Smart Card Security and Design

智能卡安全与设计

张之津　李胜广　薛艺泽　等　编著



清华大学出版社

智能卡安全与设计

张之津　李胜广　薛艺泽　等编著

清华大学出版社
北　京

内 容 简 介

智能卡已经走入平常百姓家，第二代居民身份证、手机 SIM 卡、市政公交一卡通、社保卡、电子护照、USBKey 等智能卡已经得到广泛应用，但是智能卡内含的私人信息或账户财产也越来越受到安全威胁。本书在深入介绍智能卡原理的基础上，着眼于智能卡安全内容，逐层描述了安全攻击、安全目标、安全算法、安全机制和安全规范等内容；然后介绍智能卡系统设计，包括低层设计、应用设计等内容；最后给出了智能卡未来发展的趋势。本书附有大量的研发实例。

本书的主要读者对象为从事智能卡安全开发和应用、智能卡芯片开发、芯片操作系统 COS 开发、嵌入式安全产品、机具开发和系统软件开发等工程技术人员以及高等院校相关专业师生，可作为移动通信、法定证件、市政公交一卡通、社保卡、电子政务、金融等行业专业人员的参考书籍，也可作为高校智能卡、信息安全和嵌入式等专业的本科生、研究生教材。

图书在版编目（CIP）数据

智能卡安全与设计/张之津，李胜广，薛艺泽等编著．—北京：清华大学出版社，2010.11
ISBN 978-7-302-23599-6

Ⅰ．①智…　Ⅱ．①张…　②李…　③薛…　Ⅲ．①智能卡—安全技术　②智能卡—设计
Ⅳ．①F830.46

中国版本图书馆 CIP 数据核字（2010）第 159239 号

责任编辑：王一玲
责任校对：焦丽丽
责任印制：

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62795954，jsjjc@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：185×260　　**印　　张**：22.75　　**字　　数**：553 千字
版　　次：2010 年 11 月第 1 版　　**印　　次**：2010 年 11 月第1 次印刷
印　　数：1～　000
定　　价：　.00 元

产品编号：



前言

智能卡已经走入平常百姓家，第二代居民身份证、手机 SIM 卡、市政公交一卡通、社保卡、电子护照、USBKey 等智能卡应用已经不再是新生事物，并已被广泛接受。

21 世纪初的近十年里，智能卡市场在全球稳步发展，智能卡的应用涉及了各行各业。然而平静的智能卡行业的湖面被第 24 届黑客大会（Chaos Communication Congress）投进了一块巨石，激起了世界级的安全波浪。

2008 年 4 月，在第 24 届黑客大会上，德国学者 Henry Plotz 和弗吉尼亚大学博士 Karsten Nohl 公开了全球广泛应用的 Mifare 智能卡芯片加密算法的破译方法。同年 5 月，内嵌 Mifare 芯片的伦敦公交卡被成功克隆。该事件掀起一场大范围的安全风暴，引起全球对智能卡安全的广泛关注。

中国智能卡行业深受该风暴的影响。工业和信息产业部发布了《关于做好应对部分 IC 卡出现严重安全漏洞的通知》，要求各地机关和部门开展对智能卡使用情况的调查和应对工作。智能卡专家频繁奔波于各行各业的智能卡应用系统安全评审会议中，智能卡芯片提供商不停地向用户和行业主管领导进行安全方面的解释和保证。

一张张卡片，如何让老百姓放心地使用，不必担心隐私信息的泄露，不必恐慌财产的损失。这是智能卡设计者势必每时每刻都要关心的问题。

我们必须时刻谨记：智能卡应用中，没有绝对安全的芯片，没有绝对安全的算法。任何智能卡（包括物理芯片和上层算法）都可能被攻破。因此，如何延长攻击时间、增加攻击代价或者将攻击损失降到最低是智能卡安全设计的最大理念。

正是基于以上背景，作者萌生了编写《智能卡安全与设计》这本书的想法。作者不仅是长期从事智能卡研发的科技人员，也是公安部第一研究所的专业技术警察，担负着公共安全领域的科技研究工作。

公安部第一研究所是公安部直属的警察建制的多学科、科工贸一体的综合性研究所，始建于 1960 年，主要从事警用电子设备和社会公共安全器材的研制及生产。研究所现有员工 2300 多名，其中技术人员 650 多名，包括研究员 40 余人，副研究员 150 余人；涉及警用通信、安防安检、法定证件、信息安全等多个专业。自 1993 年起，公安部第一研究所开始组织第二代居民身份证的研制工作，2004 年第二代居民身份证正式发行。自 2005 年起，研究所又承担了我国电子护照的芯片操作系统、生物特征、安全架构和一体机等全方位多层次的技术研发工作。

从事本书编写的所有作者均是公安部第一研究所法定证件领域的研究人员，在第二代居民身份证和电子护照项目研发过程中，积累了大量的实践经验。本书编写紧密联系研发经验，注重科研理论与工程实践并重，摒弃过时陈旧理论，着重未来技术应用，并附加详细开发范例供读者参考。

本书内容总共分12章，分成两大部分：智能卡安全研究和智能卡系统设计。

本书从智能卡基础（第1章）引出话题，然后按照智能卡应用的安全层次的顺序，编写了安全攻击（第2章）、安全目标（第3章）、安全算法（第4章）、安全机制（第5章）、生物特征识别（第6章）、安全规范（第7章）。

智能卡系统设计包括低层设计（第8章）、CSP应用与开发（第9章）、应用系统设计（第10章）和系统测试（第11章）。

最后给出未来智能卡发展的趋势和技术（第12章）。每章都附有详细的参考文献，为读者深入研究相关专题提供了资料。

本书由张之津研究员、李胜广博士和薛艺泽主编，其中第1、2章由李莉编写，第3、7章及11.7节由薛艺泽编写，第4章、5章、7.5节及第8章由李胜广编写，第6、10章由孙健编写，第9、11章由张小波编写，10.3节、第12章和附录由朱元硕编写。李胜广博士对全书进行了详细的审校。由于作者水平有限，书中难免存在缺点和不足，真诚地欢迎广大读者批评指正（意见或者建议请发至E-mail：lishengg@163.com）。

在本书编写过程中，公安部第一研究所厉剑所长、于锐所长助理都给予了大力支持和精心指导。科研处、证件部的同事们提供了许多技术素材，作者在此致以最诚挚的感谢！

中电智能卡、中电华大、上海华虹、NXP、Infineon、ST、握奇数据、北京航空航天大学等友好单位为本书的编写提供了技术上的支持和帮助，作者在此一并感谢！

感谢清华大学出版社王一玲主任的卓有成效的工作，使得本书得以顺利出版。

本书的出版得到了国家科技支撑计划项目“国家法定身份证件关键技术研究与应用示范”（课题编号2007BAK25B00）的支持，在此表示感谢。

作　者

公安部第一研究所

2010年4月

目录



智能卡基础

智能卡(smart card)又称集成电路卡,即 IC 卡(integrated circuit card)。它将一个集成电路芯片镶嵌在某种材质中,封装成卡式、本式或者其他形式。目前智能卡的应用已经进入高峰发展时期,智能卡广泛应用在法定证件、电信(电话卡、GSM 卡、3G 卡)、医疗保健、娱乐、公交、门票、门禁、识别、银行、有线电视节目收视收费和顾客消费等领域。典型智能卡应用如图 1-1 所示。



图 1-1　典型智能卡应用

智能卡是随着半导体技术的发展和社会对信息安全性等要求的日益提高应运而生的,它里面所包含的集成电路芯片具备微处理器及大容量存储器,具有存储、加密及数据处理能

力，被公认为世界上最小的个人计算机。与目前仍在广泛应用的磁卡相比，智能卡具有安全性高、存储容量大等许多优点，可承载比磁卡多达数百倍的信息，并能与终端结合进行复杂的计算。这种既具有智能性，又便于携带的卡片，为现代信息的处理和传递提供了一种全新手段。智能卡凭借其存储量大、可靠性强、安全性高等优点，风靡全球。

1.1 智能卡发展

智能卡的概念最初由法国人 Roland Moreno 在 1972 年提出，此后法国 Bull 公司率先投入了对这一潜力无穷的高新技术产品的研究和开发。1976 年 Bull 公司高级研究员 Ugon 先生领导的研究小组首先研制成了世界上第一张由双晶片(微处理器和存储器)组成的智能卡，接着又于 1978 年制成了单晶片智能卡并取得了技术专利。

在 20 世纪 80 年代初期，法国和德国开始了最初的智能卡应用实验。除了法国的 Bull 以外，先后有 NXP、Infineon、ST、Motorola、Sharp、Atmel、Samsung 等十几家公司相继投入了智能卡芯片和卡片成品的开发与生产，形成了一个世界性的新兴技术产业。VISA、MasterCard、EuroPay(2001 年与 MasterCard 组织合并)等三大国际信用卡组织相继推出了智能卡产品，在美洲、欧洲及亚洲的许多国家得到了推广和应用，并在各地的信用卡市场上占据了一定的份额。

中国的智能卡市场发展迅猛，各行业用卡情况简要描述如下：

2005 年始发的中国第二代居民身份证至今已累计发行 10 亿张。第二代居民身份证采用非接触式 IC 卡，集成了个人安全数据的存储和数字防伪技术，具有高安全性和可机读性。

移动手机用户已达 7.38 亿人，仅 2009 年的移动电话卡采购量就已超过 8 亿张。移动电话卡有多种分类方法：按容量可分为 16KB、32KB、64KB、128KB 甚至更高；按照无线技术可分为 GSM 网络的 SIM(subscriber identity module，用户身份识别模块)卡、CDMA 网络的 UIM(user identity module，用户识别模块)卡、3G 网络的 USIM(universal subscriber identity module，全球用户识别模块)卡。

中国各地市政公交一卡通飞速发展。市政公交一卡通是在城市公共交通应用环境中(包括公交、地铁、轻轨、出租车)，采用统一发行的非接触式 IC 卡介质作为城市公共交通储值卡，在城市不同公共交通工具上实现统一支付，并按照协定的商务规则，由统一建设的清算管理中心完成对应交通费用的结算和划转，实现无现金电子支付。目前仅北京市政公交一卡通持有量已超过 3200 万张。

中国电子护照是继中国第二代居民身份证后最大的法定证件智能卡应用项目。目前中国已签发的普通护照超过 3000 万本。预计每年新发的电子护照约 650 万份。基于安全芯片的电子旅行证件，具有高安全、高速度的特点，利用其存储的数据，不仅可以进行可靠的个人身份识别，还对打击犯罪和恐怖活动、维护国家的安全具有非常重要的意义。

美国的“智慧地球”、中国的“感知中国”又一次把物联网的概念传遍世界范围。物联网中非常重要的技术是 RFID 电子标签技术。据 ABI Research 市场技术研究公司于 2009 年 11 月统计：2009 年从 RFID 标签、读写器、软件和服务等方面获得的收入估计突破 56 亿美元，涨幅比 2008 年提高 4.25%。

为积极应对银行卡犯罪，全球 EMV 迁移战略规划正在实施中。尽管我国银行卡从磁

条卡向智能卡迁移的步伐因为技术和设备升级的巨额成本而进程缓慢，但毕竟这是一种历史的趋势。央行批准的试点城市——宁波正在推广的市民卡已带有金融应用功能，除了日常支付功能的电子钱包外，还可以直接到 ATM 上取现。中国农业银行天津市分行也发行了带有接触式智能卡的金穗卡。

其他如社保卡、医疗卡、居住证、市政公交一卡通、校园卡、旅游票卡等智能卡应用也正在稳步发展。

据国家金卡工程协调领导小组办公室统计，截止到 2009 年 12 月 23 日，全国各类智能卡发行量已达 70 亿张，在电信、交通、公安、社会保障、医疗卫生等领域得到普及应用。城市市民卡和手机支付等多功能智能卡与智能标签的应用正在创新发展中。

1.2 智能卡分类

智能卡可根据不同方式进行分类：

按照内嵌芯片的电路结构不同，智能卡可分为非加密存储卡（memory card）、逻辑加密卡（memory card with security logic）和 CPU 卡（smart card）。

（1）非加密存储卡内嵌芯片为存储卡芯片，相当于普通串行 EEPROM（电可擦写可编程只读存储器），有些芯片还增加了特定区域的写保护功能。这类卡信息存储方便，使用简单，价格便宜，很多场合可替代磁卡。但由于其本身不具备信息保密功能，因此，只能应用于保密性要求不高的场合。

（2）逻辑存储卡是在非加密存储卡的基础上增加了加密逻辑电路，加密逻辑电路通过校验密码的方式来判断卡内的数据对于外部访问是否开放，这样提高了卡的保密性和安全性。但这只是低层次的安全保护，无法防范恶意攻击。

（3）CPU 卡内的集成电路带有微处理器（CPU）、存储单元（包括随机存储器（RAM）、只读存储器（ROM）、电可擦除存储器（EEPROM））以及芯片操作系统（chip operating system，COS）。装有 COS 的 CPU 卡不仅具有数据存储功能，同时具有命令处理和数据安全保护等功能。由于 CPU 卡具有存储容量大、处理能力强、信息存储安全等特性，所以被广泛应用于信息安全性要求特别高的场合。本书后面提到的智能卡如不特殊说明，均指 CPU 卡。

按照封装方式，智能卡又可分为卡式、本式或者其他方式。

按照数据 I/O 接口方式，智能卡可分为接触式智能卡、非接触式智能卡、光通信和 USB 接口智能卡。

不同的分类方式如图 1-2 所示。

本节主要按照数据读写方式进行重点描述。其中，接触式智能卡由读写设备的触点和卡片上的触点相接触，进行数据读写。非接触式智能卡则与读写设备无电路接触，由非接触式的读写技术进行读写（如光通信或无线射频通信）。其内嵌芯片除了存储单元、控制逻辑外，增加了射频收发电路。而同时具有接触式和非接触式接口的卡片称为双界面智能卡。双界面智能卡同时具有射频天线和触点，两者互不影响，两种通信接口的组合扩展了智能卡的应用领域。

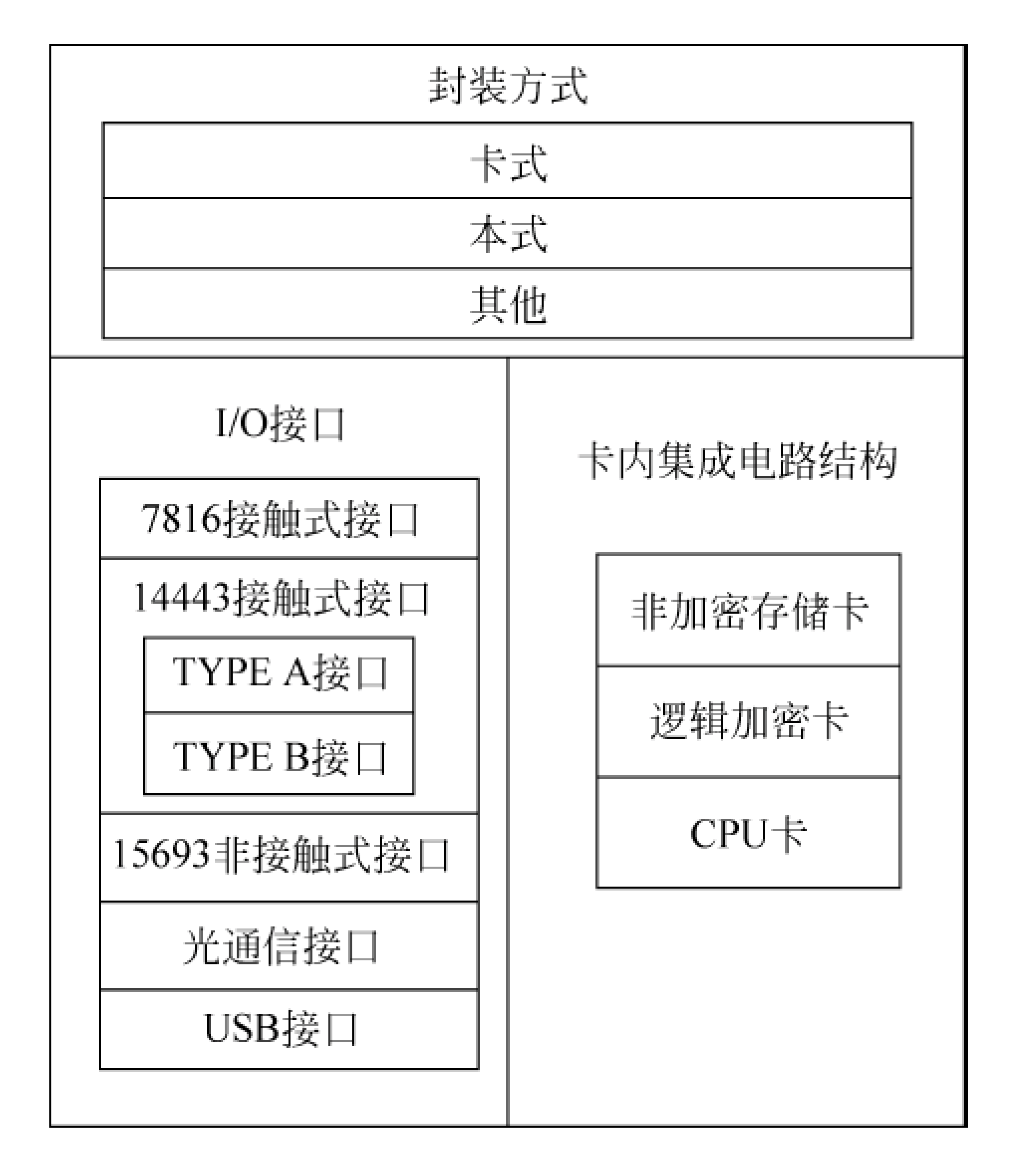


图 1-2 智能卡分类

1.2.1 接触式 IC 卡

接触式 IC 卡片的外形尺寸与基本结构遵循 ISO/IEC 7810 标准，具体参数如图 1-3 所示。

接触式 IC 卡(contact card)通过一组 6～8 个金属触点(如图 1-4 所示)，建立与外界的接口，由读写器的接触弹簧和 IC 卡上的触点产生电流接触，读写器通过接触触点给 IC 卡提供能量和时钟脉冲，读写器与 IC 卡之间的数据传输是通过双向串行接口(I/O)进行。接触式 IC 卡系统由卡基(塑料卡片)、触点、芯片、读写器等组成。

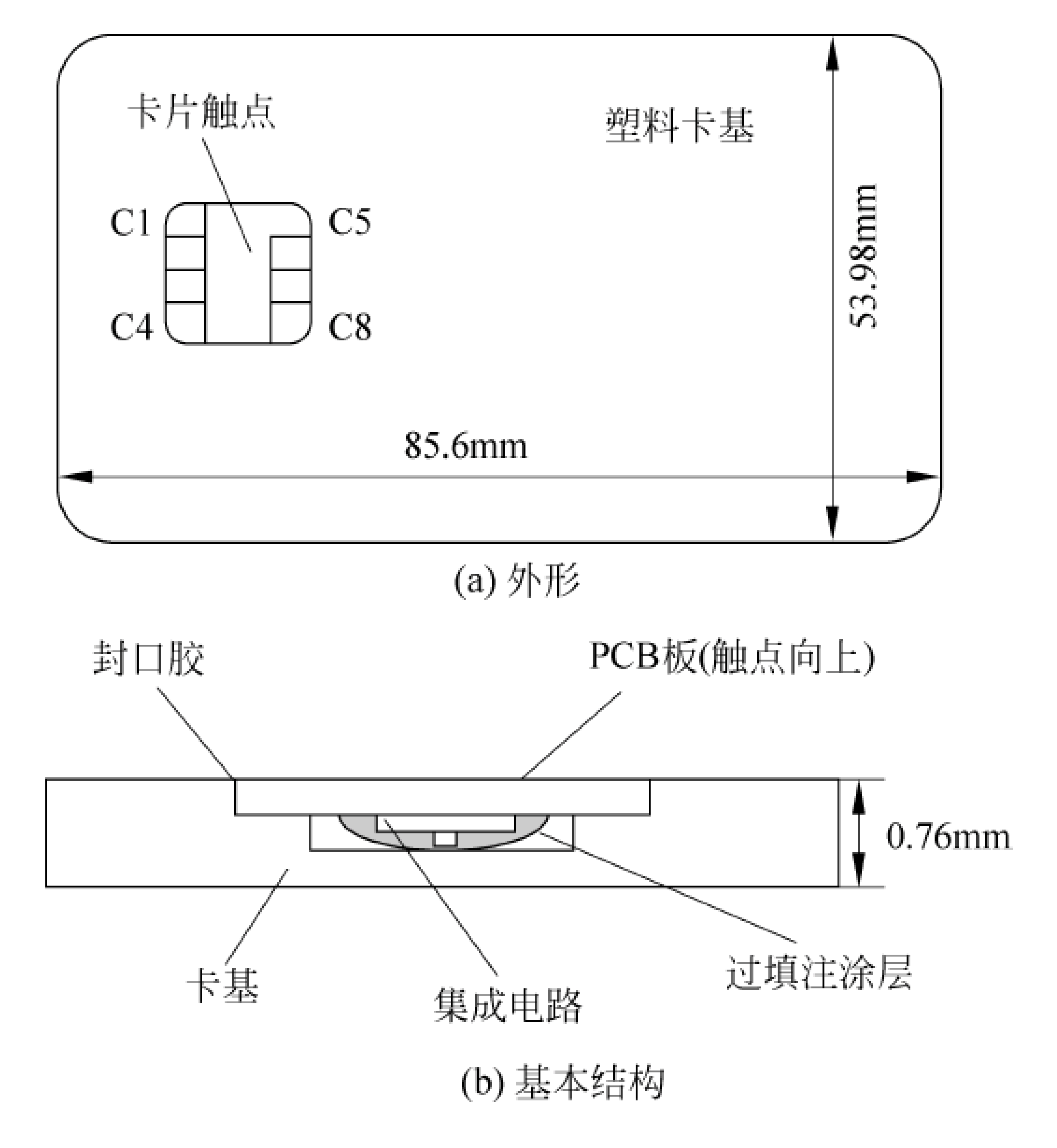


图 1-3 接触式 IC 卡的外形与基本结构

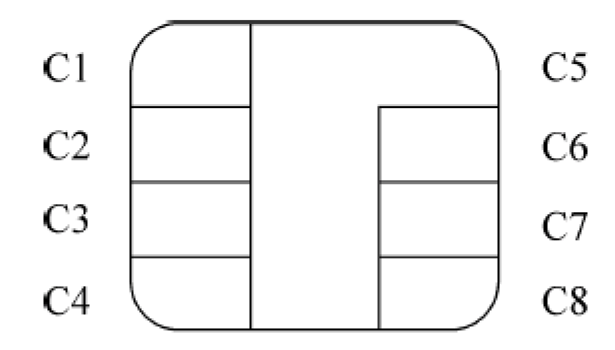


图 1-4 IC 卡触点定义

ISO/IEC 7816-2 对接触式卡上的触点有明确规定，具体规定如表 1-1 所示。

表 1-1 IC 卡触点功能定义

触点	功　能	触点	功　能
C1	VCC(工作电源)	C5	GND(接地)
C2	RST(复位信号)	C6	VPP(编程电源)
C3	CLK(时钟)	C7	I/O(数据输入输出)
C4	RFU(待用)	C8	RFU(待用)

1.2.2 非接触式 IC 卡

非接触式 IC 卡(contactless card)与读写设备不通过机械接触，而是利用射频技术(radio frequency identification，RFID)，通过电磁波传输进行 IC 卡的数据读写。根据工作频率的不同分为低频、中频、高频三种；按卡内有无电源分为有源和无源两种，目前大多设计采用无源方式。卡内能量的来源由天线接受射频脉冲，整流并给电容充电，电容电压经过稳压后作为工作电源。非接触式卡片主要由芯片和天线组成，连接方式如图 1-5 所示。

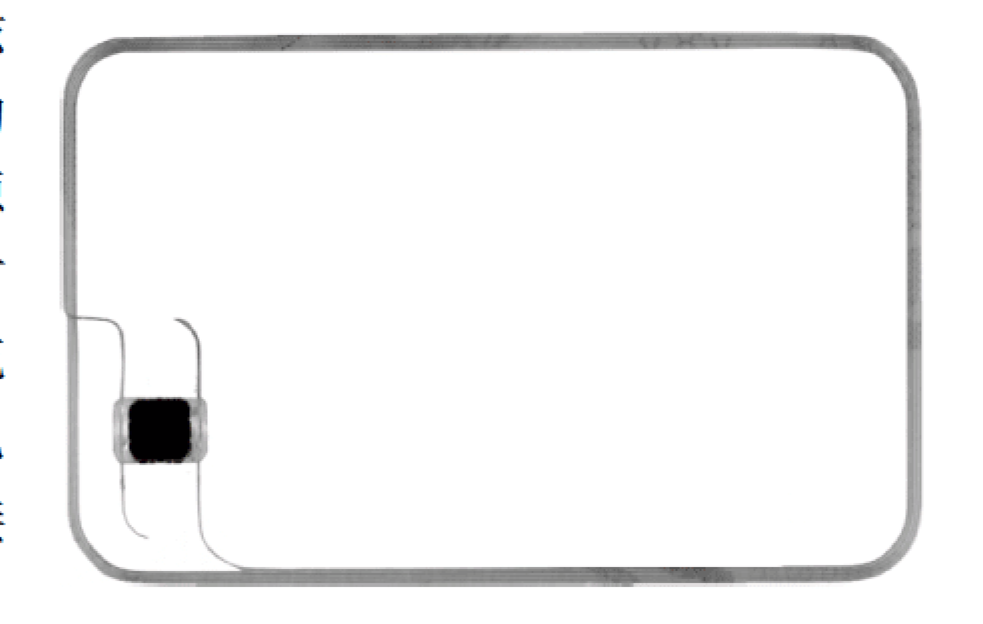


图 1-5 非接触式 IC 卡芯片和天线

非接触式 IC 卡读写系统工作原理：系统以半双工方式在读写器与 IC 卡之间双向传递数据。读写器将要发送的信号编码后，加载在特定频率的载波信号上，经天线向外发送；进入读写器工作区域的 IC 卡接收到脉冲信号后，一方面芯片中的射频接口模块由此信号获得电源电压和复位、时钟信号，同时该芯片中的有关电路对此信号进行调制、解码、解密，然后对命令请求、密码、权限等进行判断，控制逻辑电路从存储器中读取有关信息，经加密、编码、调制后经卡内天线发送给读写器处理。

1.2.3 双界面卡

所谓双界面卡(combi-card)，就是将接触式接口和非接触式接口集合在同一实体上的 IC 卡。非接触式 IC 卡在通信方式上不用电触点，因此其拥有比接触式 IC 卡优越的便捷性、低故障率和高环境耐受能力，但电磁信道削弱了其对高强度射频干扰的抑制能力，该信道又极易成为窃听和篡改等非法攻击的入侵点，所以在高安全性应用中，非接触式 IC 卡的可信度令人担忧。另一方面，IC 卡应用的两个重要领域——金融和电信行业都已经建立大量接触式 IC 卡基础设施，并经过多年实践考验，短期内不可能弃而不用。因此，将非接触式 IC 卡的使用的方便性和接触式 IC 卡的安全性融于一体，双界面卡方案是一种技术要求较高，但也是最成功并且得以广泛应用的方案。

双界面卡有以下两类：

(1) 接触式智能卡系统与非接触式智能卡系统彼此操作独立，但共享卡内部分存储空间(如 EEPROM)，示意图参见图 1-6。

(2) 接触式智能卡系统与非接触式智能卡系统完全融合,接触式与非接触式运行状态相同,共用一个 CPU 管理,如图 1-7 所示。

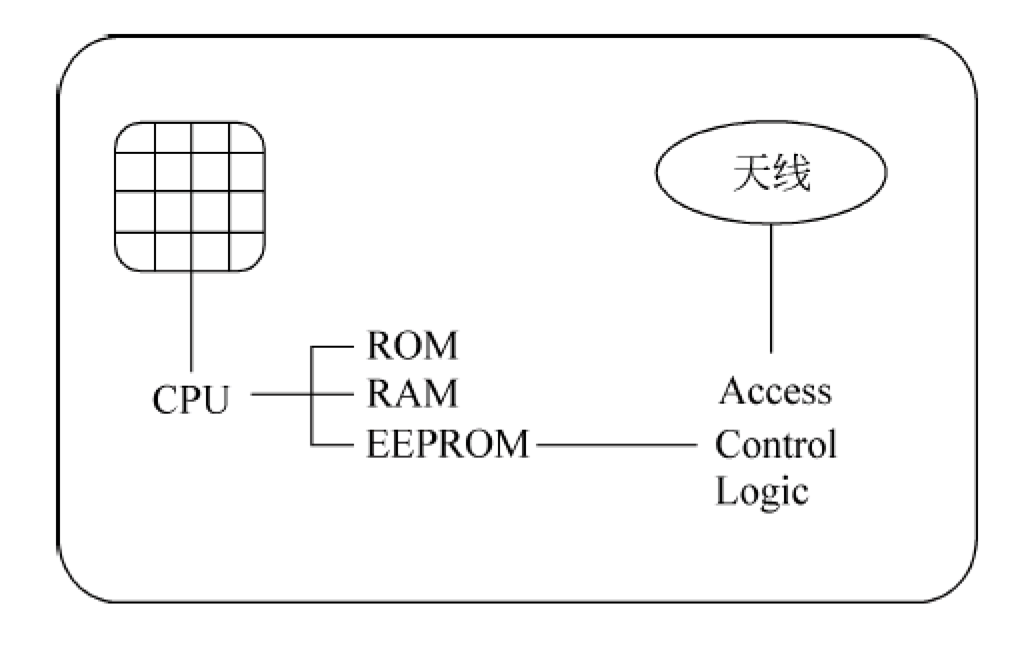


图 1-6 双界面卡图示 1

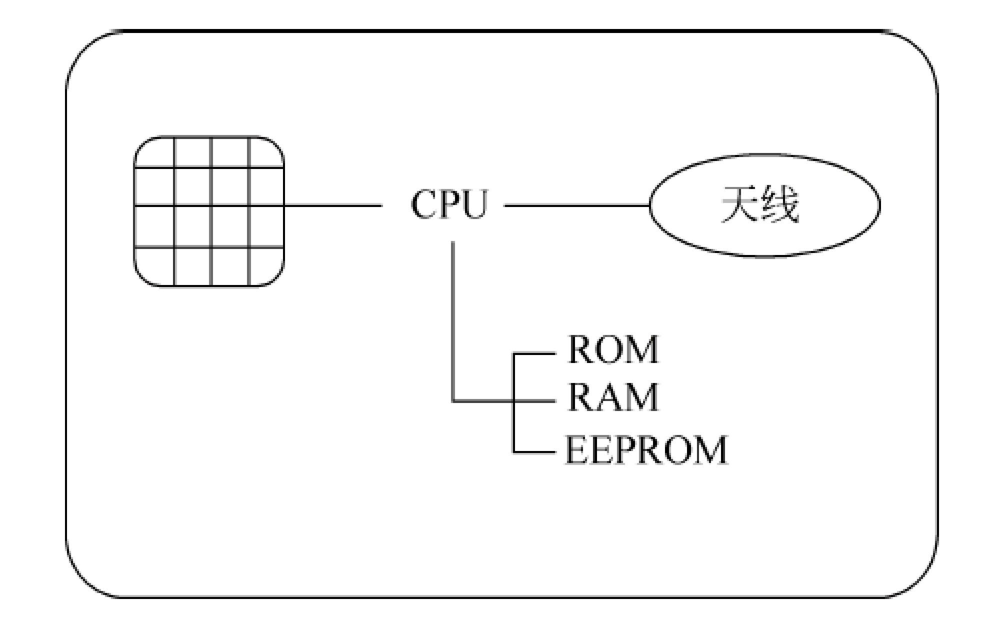


图 1-7 双界面卡图示 2

1.2.4 其他类卡

其他类卡是在 ISO 标准中没有涉及或暂时没有规定的,但已经研制出来的卡,它们改进或应用了 IC 卡的芯片技术,主要包括以下几类:

(1) 多媒体 IC 卡。外形尺寸小,容量大。

(2) P 型 IC 卡。具有大容量高速度的特点,为并行通信 IC 卡。

(3) VHDR(very high data rate)卡。这种卡片通信速度可达到 10Mb/s。

(4) 异型卡。把芯片制作在不规则外形的材料中,如钥匙型卡、USB 设备、纽扣型卡等,可以用作身份识别、安全认证或加密等场合。

1.2.5 性能比较

接触式 IC 卡的技术比较成熟,与磁卡相比,具有存储容量大、体积小、防磁、抗干扰能力强、安全性高、对网络要求不高、可以一卡多用等特点。但它也存在自身的缺点:机械接触易磨损,不能抵御灰尘、油污等外界恶劣环境影响。非接触式 IC 卡利用射频技术,有效地解决了接触式 IC 卡的接触磨损和受使用环境所限的缺点,但存在高速的非接触式 CPU 卡如何供电的问题和因电流微弱而难以进行高性能的加密处理,以及不需要接触就可以交易带来的客户无意识交易等问题。然而非接触式 IC 卡技术的兴起,还是给 IC 卡的应用带来了革命,使 IC 卡的应用空前广泛起来。

对常见的条码卡、磁条卡、接触式 IC 卡、非接触式 IC 卡等识别技术而言,在数据容量、读写方式、可否重写等方面,它们都有各自的特点及适于应用的场合,如表 1-2 所示。

表 1-2 常见识别技术的比较

技术性能	条码卡	磁卡	接触式 IC 卡	非接触式 IC 卡
数据容量	小	较小	大	大
读取方式	光电转换	电磁转换	金属触点	无线通信
可否重写	否	可	可	可
一卡多用	否	否	可	可

续表

技术性能	条码卡	磁卡	接触式 IC 卡	非接触式 IC 卡
保密性	差	差	好	好
使用寿命	短	短	长	最长
读写稳定性	不高	不高	不高	高
机械故障率	低	高	高	低
操作简便性	一般	一般	一般	简便
耐用性	差	差	一般	好

1.3 智能卡规范

接触式 IC 卡遵循的是 ISO/IEC 7816 国际标准，非接触式 IC 卡的国际标准为 ISO/IEC 10536、ISO/IEC 14443、ISO/IEC 15693，以及 ISO/IEC 7816 中对非接触式 IC 卡也使用的部分标准。其中 ISO/IEC 10536 目前已基本不用。

ISO/IEC 10373 是有关识别卡（包括磁卡、光卡、IC 卡）测试的国际标准。

1.3.1 接触式 IC 卡规范

ISO/IEC 7816 国际标准的标题是：识别卡—集成电路卡。

该标准包括以下部分：

- ISO/IEC 7816-1：接触式卡的物理特性。
- ISO/IEC 7816-2：触点尺寸和位置。
- ISO/IEC 7816-3：异步卡的电接口和传输协议。
- ISO/IEC 7816-4：组织、安全和用于交换的命令。
- ISO/IEC 7816-5：应用提供者的注册。
- ISO/IEC 7816-6：用于交换的数据元。
- ISO/IEC 7816-7：结构化卡查询命令语言。
- ISO/IEC 7816-8：安全操作命令。
- ISO/IEC 7816-9：卡管理命令。
- ISO/IEC 7816-10：同步卡的电接口和复位应答。
- ISO/IEC 7816-11：个人验证的生物方法。
- ISO/IEC 7816-12：USB 卡的电接口和操作过程。
- ISO/IEC 7816-13：在多应用环境中用于应用管理的命令。
- ISO/IEC 7816-15：密码信息应用。

1.3.2 非接触式 IC 卡规范

根据非接触式 IC 卡操作时与读写器发射表面距离的不同，定义了三种卡（CICC、PICC 和 VICC）及其相应的读写器（CCD、PCD 和 VCD），如表 1-3 所示。

表 1-3　非接触式 IC 卡、读写器及对应的国际标准

IC 卡	读写器	国际标准	读写距离
CICC	CCD	ISO/IEC 10536	紧靠
PICC	PCD	ISO/IEC 14443	＜ 10cm
VICC	VCD	ISO/IEC 15693	＜ 50cm

表中 ICC 表示集成电路卡；CICC 为 close-coupled ICC；PICC 为 proximity ICC；VICC 为 vicinity ICC；CD 为 coupling device，是读写器中发射电磁波的部分。

与接触式 IC 卡相比，非接触式 IC 卡需要解决下述 3 个问题：

(1) IC 卡如何取得工作电压。

(2) 读写器与 IC 卡之间如何交换信息。

(3) 多张卡同时进入读写器发射的能量区域(即发生冲突)时，如何处理。

非接触式 IC 卡的国际标准为 ISO/IEC 10536(基本不用)、ISO/IEC 14443、ISO/IEC 15693 等标准。在 ISO/IEC7816 标准中也有适用于非接触式 IC 卡的内容，具体描述如下。

1. ISO/IEC 14443

ISO/IEC 14443 国际标准的标题是：识别卡—非接触式集成电路卡—接近式卡。

该标准共分 4 个部分：

- ISO/IEC 14443-1：物理特性。
- ISO/IEC 14443-2：射频能量和信号接口。
- ISO/IEC 14443-3：初始化和防冲突。
- ISO/IEC 14443-4：传输协议。

2. ISO/IEC 15693

ISO/IEC 15693 国际标准的标题是：识别卡—非接触式集成电路卡—邻近式卡。

该标准共分 4 个部分：

- ISO/IEC 15693-1：物理特性。
- ISO/IEC 15693-2：空中接口和初始化。
- ISO/IEC 15693-3：防冲突和传输协议。
- ISO/IEC 15693-4：传输协议。

3. ISO/IEC 7816 中适用于非接触式 IC 卡的部分

- ISO/IEC 7816-4：组织、安全和用于交换的命令。
- ISO/IEC 7816-5：应用提供者的注册。
- ISO/IEC 7816-6：用于交换的数据元。
- ISO/IEC 7816-7：结构化卡查询命令语言。
- ISO/IEC 7816-8：安全操作命令。
- ISO/IEC 7816-9：卡管理命令。
- ISO/IEC 7816-11：个人验证的生物方法。
- ISO/IEC 7816-13：在多应用环境中用于应用管理的命令。
- ISO/IEC 7816-15：密码信息应用。

1.4 智能卡系统

1.4.1 硬件结构

智能卡由芯片和固化于芯片中的嵌入式智能卡操作系统及应用软件组成。一张智能卡即构成了一台便携和抗损的微型计算机。

智能卡的芯片结构主要包括五部分,参见图 1-8。

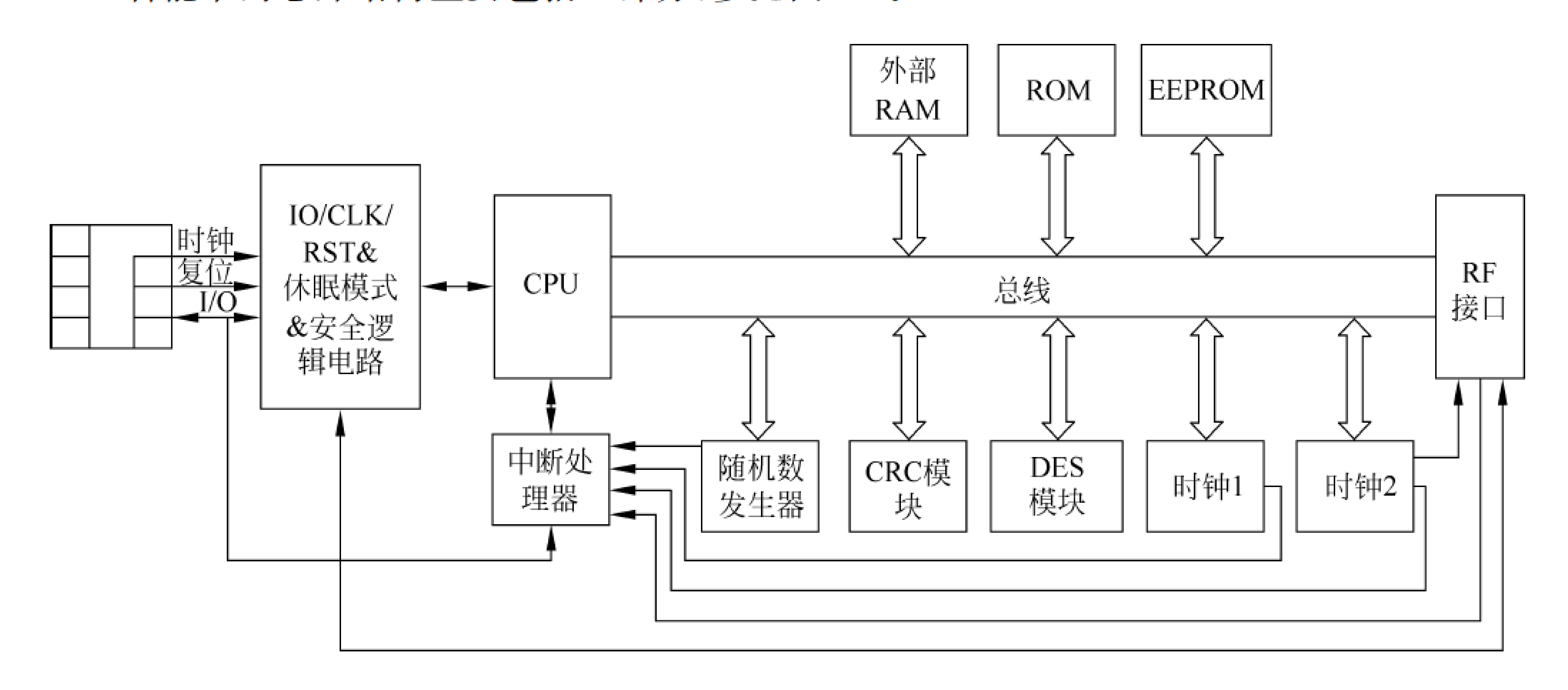


图 1-8 智能卡的芯片结构

微处理器(CPU)通常是一个 8/16/32 位的处理器,目前最常见的是 8051 和 80251,负责完成所有运算和数据交换功能。以后还将出现新的、功能更强的处理器。

随机存储器(RAM)用于存放临时数据或中间数据,例如短期密码、临时变量和堆栈数据等,容量通常比较小。

只读存储器(ROM)中固化的是操作系统代码及自测程序,其容量取决于所采用的微处理器。要注意的是,代码是通过掩模存储的,不能以任何方式更改。

电可擦除可编程只读存储器(EEPROM)中存储了智能卡的各种应用信息,如加密数据和应用文件等,有时还包括部分 COS 代码,容量通常介于 1～144KB,这部分存储资源可供用户开发利用。大容量智能卡则一般使用 FLASH 存储介质存储数据,容量也在兆级。

通信器件用于在智能卡和外部访问终端之间交换数据和控制信息,可分为接触式和非接触式接口。

1.4.2 芯片操作系统

智能卡是数据信息的安全载体,基本功能为数据的存储和安全保护。芯片操作系统(chip operating system,COS)是嵌入在智能卡芯片内,管理硬件资源、执行数据的安全存取并与外部接口设备通信的监控软件系统。

COS属于嵌入式定制软件，需依赖于所嵌入的智能卡芯片。不同的智能卡芯片，COS系统也不尽相同。国际标准仅规范卡片的外部尺寸、硬件特性和不同厂商间的通信命令，并不限制COS软件的设计架构和理念。COS软件设计理念除了具有普通软件开发的普遍性，还具有行业专用性、代码高效性、资源功耗紧缩性等嵌入式软件特点。

1.4.3 系统构件安全

智能卡应用系统如图1-9所示，从底层往上依次为智能卡、机具、驱动和相关服务、应用程序，其中芯片属于专业公司设计和制造，有专业芯片手册进行介绍，所以其内容不在本书讨论范围内。

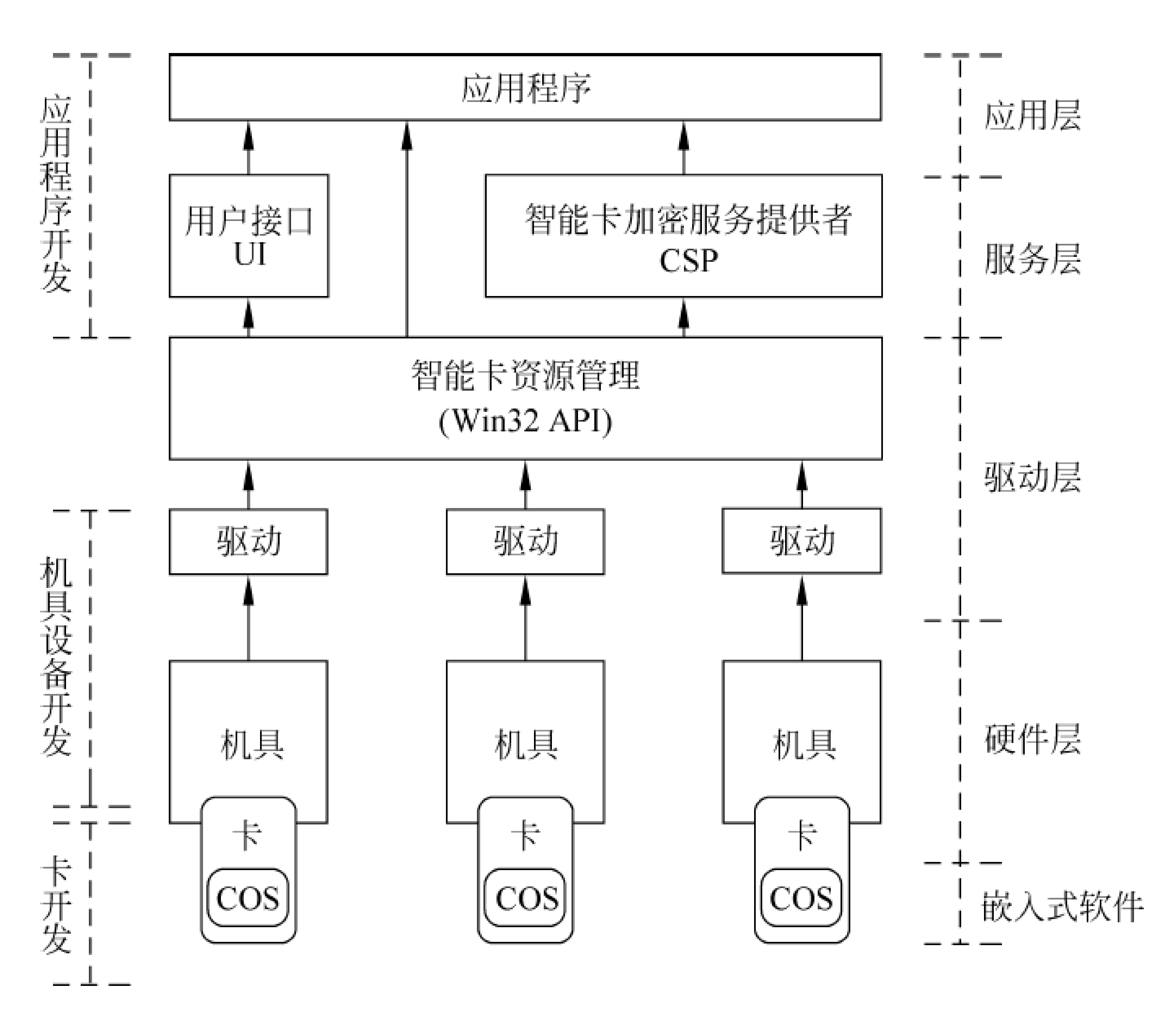


图1-9 智能卡应用系统框架图

本书的主要研究对象是整个智能卡应用系统中各个层次的构件安全和层次间的通信安全。构件安全包括底层智能卡的安全、机具的安全和应用的安全；通信安全则包括智能卡与机具之间的通信安全、机具与应用之间的通信安全。

底层智能卡安全又包括芯片安全和COS安全。智能卡芯片安全包括在芯片设计过程中，提高芯片设计的复杂程度和芯片制造的精细程度。智能卡内嵌COS系统针对数据的安全存储和授权访问涉及多种安全算法，如3DES、AES等对称加密算法，RSA、ECC等非对称加密算法。

智能卡与机具之间数据传输的安全问题是指攻击者对智能卡与接口设备之间通信线路中的信息流进行截听、复制或者修改。安全通信则重点抵御通信链路过程中受到的各种攻击。通常采用多种安全算法和安全机制保证智能卡与机具之间的通信安全。

机具和应用的安全除了常规软硬件安全外，还利用CSP等服务、生物特征识别机制来实现智能卡业务的安全服务。

1.5 本书结构

本书共有 12 章，分别介绍了智能卡的安全攻击、安全目标、安全算法、安全机制、生物特征识别、安全规范、系统设计、安全测试、典型应用等。本书结构参见图 1-10。

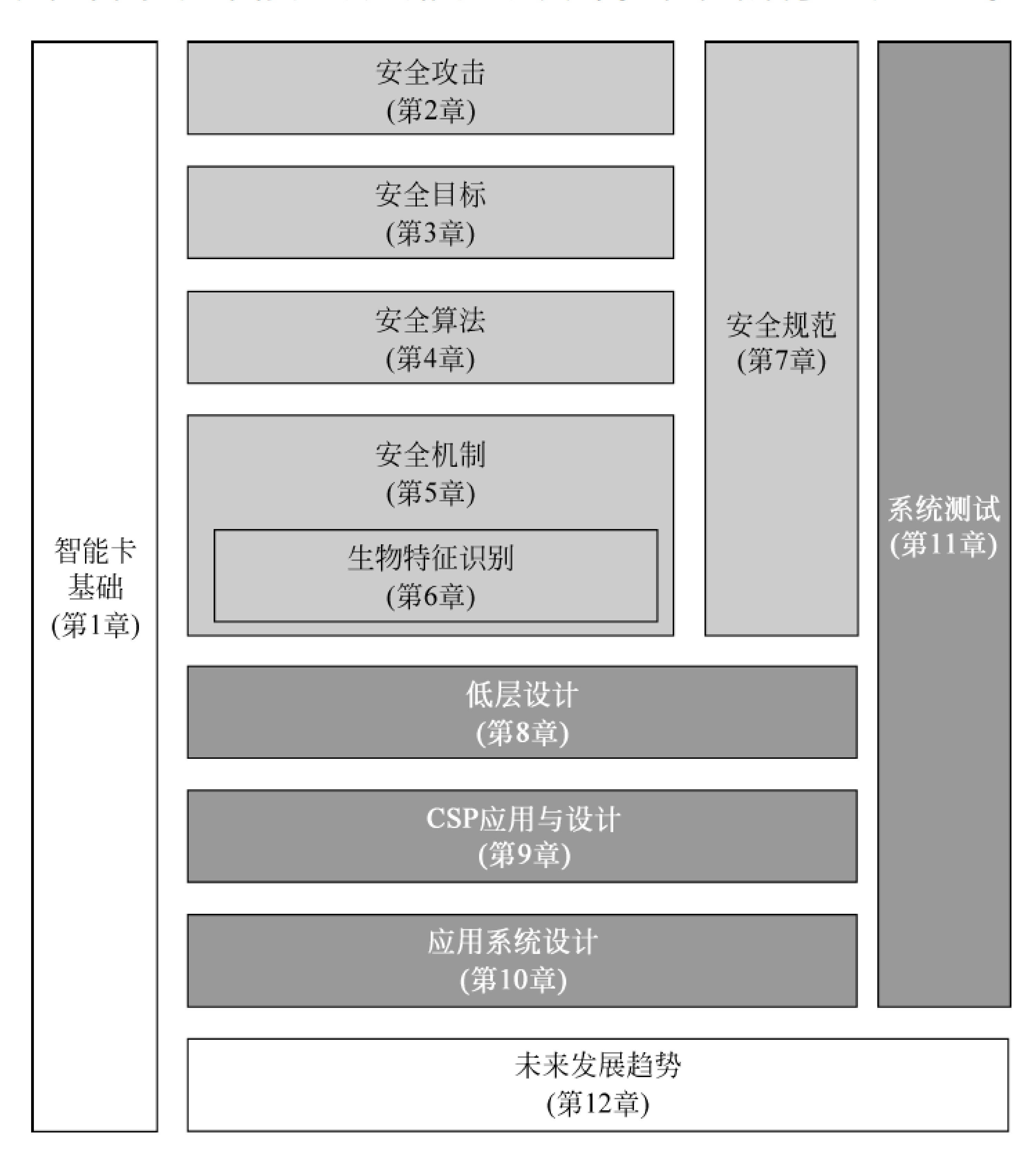


图 1-10 本书结构图

第 1 章智能卡基础，主要介绍智能卡发展、分类和规范，详细描述了智能卡系统的层次划分。

第 2 章安全攻击，介绍智能卡硬件反向工程攻击、通信链路攻击和 COS 逻辑攻击，并简单给出在芯片设计和应用设计时应注意的安全策略。

第 3 章安全目标，介绍整个智能卡系统的安全目标，包括智能卡芯片的安全目标、芯片操作系统的安全目标以及智能卡应用环境的安全目标。

第 4 章安全算法，介绍用于智能卡内嵌 COS 系统中的对称算法 DES/3DES/AES，非对称算法 RSA/ECC，散列算法 SHAx 和消息认证码 MAC 算法。并给出了计算范例。

第 5 章安全机制，介绍用于智能卡内嵌 COS 系统中并基于安全算法的安全机制，主要为智能卡数据完整性、机密性和不可否认性等安全要求服务。

第 6 章生物特征识别，介绍智能卡应用业务中生物特征识别的流程、算法。生物识别技术成为弥补密钥认证机制的最佳方法，在智能卡应用中得到了广泛的应用。

第 7 章安全规范，介绍应用于智能卡的与安全有关的国际规范，并简要介绍了国际标准化组织概况。

第 8 章低层设计，介绍智能卡系统层次划分，主要描述了芯片操作系统的设计、接触式/非接触式读写设备的设计和应用程序的设计。

第 9 章 CSP 应用与开发，介绍在开发智能卡应用系统过程中涉及的加密服务提供者接口，也介绍了智能卡变体 USBKey 作为 CSP 的设计方法。

第 10 章应用系统设计，介绍智能卡在城市交通、物流、可信计算、电子商务、3G 电信业务等行业的应用系统方案设计。

第 11 章系统测试，介绍智能卡系统在可用性、可靠性和安全性等多方面、多层次的测试。并给出了安全测评流程和相关业务介绍。

第 12 章未来发展趋势，介绍趋于一统的智能卡规范、未来的智能卡发展方向以及物联网技术，探索智能卡领域未来发展的趋势。

附录中给出了辅助本书正文的代码和工具软件说明等内容。

参考文献

[1] International Standard ISO/IEC 7816-4. Identification cards. Integrated circuit(s) cards，Part4：Organization，security and commands for interchange，2005

[2] EMV2000. Integrated Circuit Card. Specification for Payment Systems. Book 2-Security and Key Management，2000

[3] International Standard ISO/IEC 14443-3. Identification Cards. Contactless Integrated Circuit(s) Cards. Proximity Cards，Part 3：Initialization and Anti-collision，2002

[4] International Standard ISO/IEC 14443-4. Identification Cards. Contactless Integrated Circuit(s) Cards. Proximity Cards，Part 4：Transmission Protocol，2001

[5] 张利华，朱灿焰，张其善. 智能卡及其应用技术研究. 微型机与应用，2002，21(12)：4～6

[6] 张大伟，靳伟. Java 智能卡原理与应用开发. 北京：电子工业出版社，2008

[7] 李振涛，吴松. IC 卡分类及其应用分析. 石家庄铁路工程职业技术学院学报，2002，1(2)：43～48

[8] 王爱英. 智能卡技术——IC 卡. 第 3 版. 北京：清华大学出版社，2009

[9] ISO/IEC 9797-1. Information technology. Security techniques. Message Authentication Codes (MACs)，Part 1：Mechanisms using a block cipher，1999

[10] ISO/IEC 9797-2. Information technology. Security techniques. Message Authentication Codes (MACs)，Part 2：Mechanisms using a dedicated hash-function，2002

[11] ISO/IEC 9796-2. Information technology. Security techniques. Digital signature schemes giving message recovery，Part 2：Integer factorization based mechanisms，2002

[12] ISO/IEC 9796-3. Information technology. Security techniques. Digital signature schemes giving message recovery，Part 3：Discrete logarithm based mechanisms，2000

[13] (英)亨德利著，杨义先等译. 智能卡安全与应用. 北京：人民邮电出版社，2002

[14] 李莉，刘建伟. RFID 安全保密技术研究进展. 信息安全与通信保密，2007(8)：165～167

[15] Wolfgang Rankl，Wolfgang Effing. 智能卡大全——智能卡的结构. 功能. 应用. 第 3 版. 北京：电子工业出版社，2002



安全攻击

智能卡受到的攻击按照攻击层次可分为物理层攻击、逻辑层攻击和应用层攻击。应用层攻击涉及常见的网络安全、数据库安全和软件安全，本书不再赘述。本书着重描述智能卡物理层攻击和逻辑层攻击。智能卡物理层攻击可分成物理攻击、旁路攻击和半入侵式攻击。逻辑攻击则可分成通信链路攻击、COS逻辑攻击等。

2.1 物理攻击

本节介绍针对智能卡芯片的物理攻击方法。这些攻击方法要在特殊的实验室里使用特殊的设备才能实施，而且通常需要较长时间才能完成。由于物理攻击会损坏智能卡的封装，因此也常称其为入侵式攻击。

2.1.1 物理攻击基本方法

1. 分解(de-packaging)

在实施物理攻击之前，要先将裸片从塑料卡片中取出来。具体过程是，首先用刀子切掉包裹模块的塑料，露出环氧树脂；然后使用发烟硝酸去除包裹裸片的环氧树脂；再用丙酮清洗残留的酸和环氧树脂，剩下裸露的硅片。

2. 版图重构(layout reconstruction)

将卡片分解之后，下一步就是重构目标芯片的版图。通过研究连接模式和跟踪金属连线穿越可见模块(如ROM、RAM、EEPROM、ALU、指令译码器等)的边界，可以迅速识别芯片上的一些基本结构，如数据线和地址线。

通常，芯片表面的照片只能完整显示顶层金属的连线，而且它是不透明的。攻击者需要借助高性能的成像系统，从顶部的高低不平中识别出较低层的信息。但是对于提供氧化层平坦化的CMOS工艺，还需要逐层去除金属才能进一步了解其下的各种结构。

图2-1是一个NAND门驱动一个反向器的光学版图照片，类似于该图的不同层照片对于有经验的人无异于电路图。

版图重构的技术可用于获得只读型ROM的内容。ROM的位模式存储在扩散层，用氢氟酸(HF)去除芯片各覆盖层后，根据扩散层的边缘就很容易辨认出ROM的内容，图2-2可以清晰地看到ROM内容。ROM中可能不包含任何加密的密钥信息，但是它包含足够的I/O、存取控制、加密程序等信息，这些在非入侵式攻击中尤为重要。

3. 微探针技术(micro-probing)

物理攻击中最重要的工具是微探针工作台，它主要是由一台专用光学显微镜构成。图2-3为在微探针工作台上对芯片进行攻击的示意图。

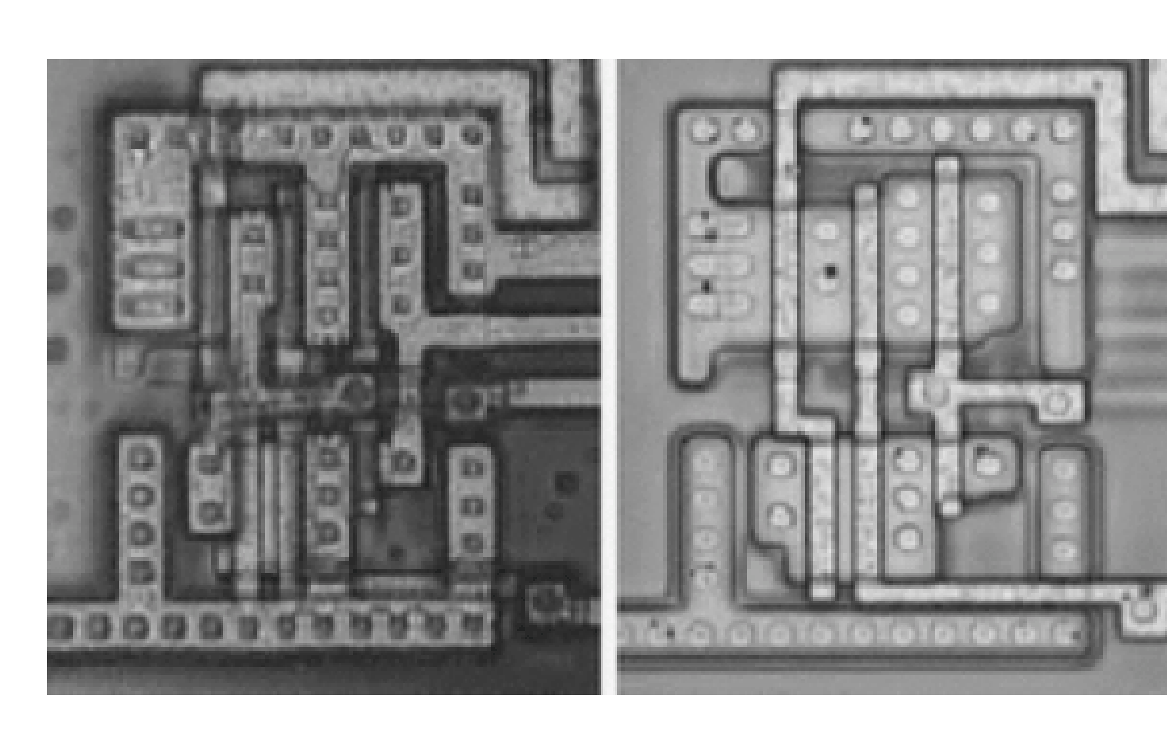


图 2-1　反向器光学版图照片

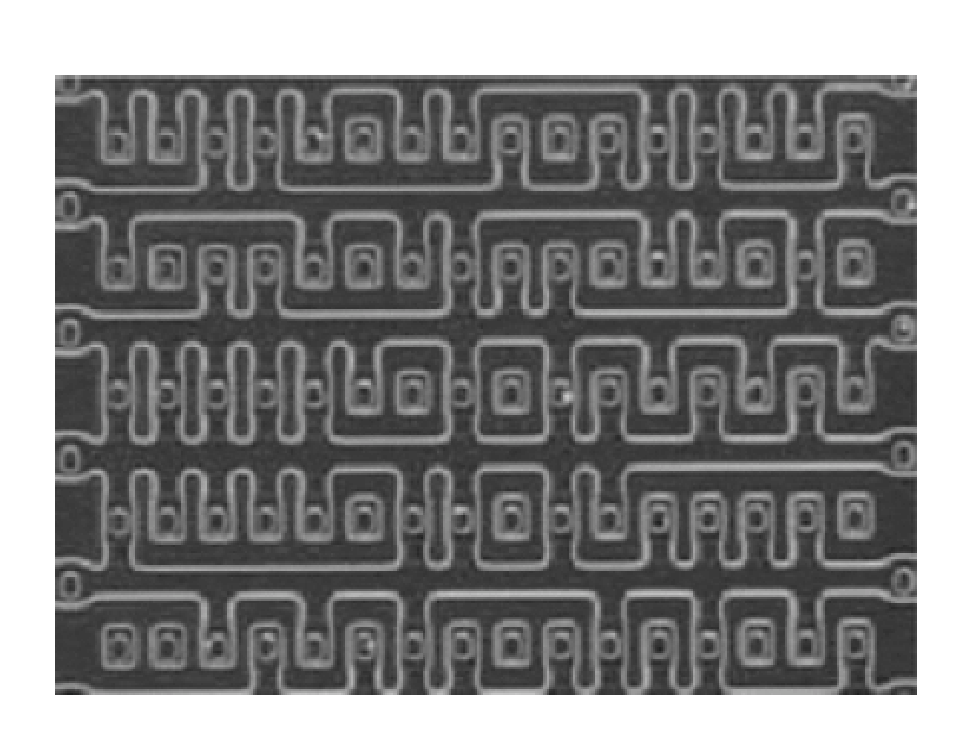


图 2-2　ROM 内容图

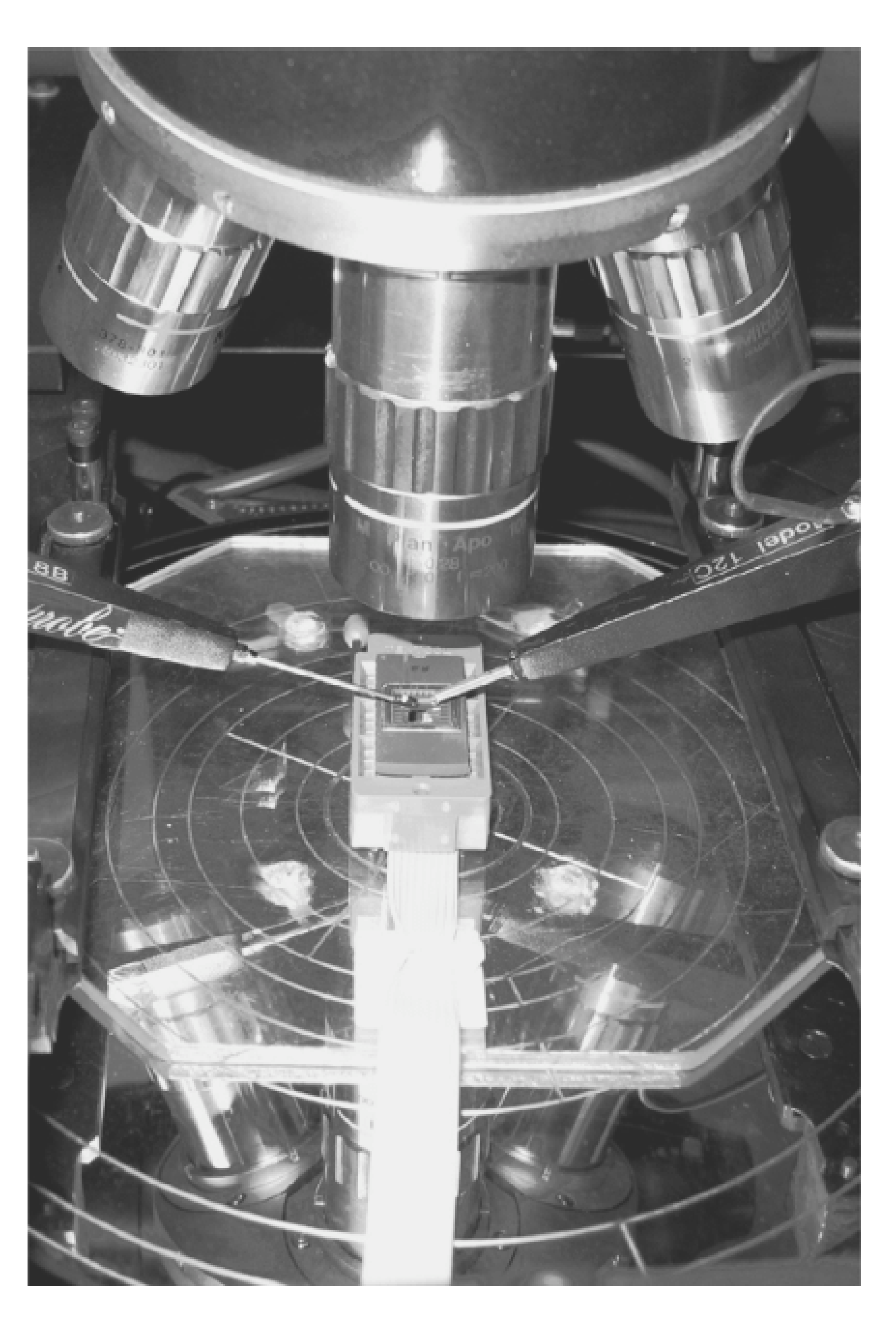


图 2-3　微探针工作台

在显微镜的一侧，攻击者安装一个钨丝探针，该探针可以在不破坏芯片功能焊盘的前提下与芯片建立电气连接。探针经过一个放大器，再连接到数字信号处理卡上，该卡记录并控制芯片处理器信号，同时也可得到芯片的能量、时钟、复位以及 I/O 信号。攻击者可以使用探针获取感兴趣的信号，从而分析出智能卡的有关设计信息和存储结构，甚至直接读取出存储器的信息进行分析。

4. 高级“束”技术(advanced beam technologies)

对具有多金属层和特征小于可见光波长的智能卡，需要使用更加昂贵的攻击工具。

1) 离子束(ion beams)

在不破坏芯片表面电路结构的情况下，用含有不同气体的粒子束，可在芯片上沉积出导线、绝缘体甚至半导体。采用这种方法可重新连接测试电路的熔断丝，或将多层芯片中深藏

在内部的信号连到芯片的表面，或加粗加强过于纤细脆弱无法置放探针的导线，从而形成一个新的“探针台”。技术人员可利用激光干涉仪工作台观察芯片单个晶圆的微细结构以及其他的电路结构。

2）电子束测试仪（electron beam testers）

在两种情况下，电子束测试仪是非常有用的攻击工具。第一种是被攻击的处理器的时钟频率能降低到100kHz以下，此时允许实时记录所有总线的信号；第二种是处理器能被强制重复执行相同操作，这样就会产生周期性的信号。

3）红外激光（infrared laser）

红外激光攻击智能卡技术原本是桑迪亚国家实验室的一项涉密技术。该技术利用一种可以穿透硅元素的红外光线对智能卡芯片进行透视，由此产生的光电流允许对该芯片的运作和单个晶体管的逻辑状态进行识别和探测。

2.1.2 存储器攻击

1. 通过总线探测读取存储区内容

存储的数据只能通过内存总线访问。微探针技术就是用来观测总线的，当存储器中的数据被访问时，它就能将其记录下来。

仅仅让处理器重复某些操作还不足以遍历存储器的所有位置。有一种智能卡的设计思路是对智能卡的每一次重置都计算和验证存储器校验值，设计者认为该操作能提高智能卡的抗干扰能力。当然，这样的设计也给攻击者提供了接近整个存储区的机会，简化了读取整个存储区内容的难度。

为了在不借助智能卡软件的情况下读取存储单元的数据，攻击者必须控制CPU的一部分，比如程序计数器。程序计数器在每个指令周期都会自动增加，并被用来读取下一个地址，这使得它非常适合做一个地址序列发生器。攻击者必须防止处理器执行jump、call、return指令，因为这些指令会扰乱程序计数器的正常记录顺序。用激光对指令解码器或程序计数器电路做微小的修改是很容易实现的，而这常常能收到很好的效果。

2. 恢复和使用测试态

对于一般意义的集成电路产业链条来说，需要将不良的芯片在晶圆测试阶段剔除，以减少后序加工工序中不必要的浪费。晶圆测试内容包括性能测试、逻辑功能测试和存储器测试。如果借助于芯片应用功能来进行片上逻辑和存储器测试，则测试成本将大幅增加，所以通常采取等效测试原理设计额外的测试态来快速完成测试工作。

图2-4(a)是在接触式智能卡芯片的发展过程中曾大量采用的测试态控制方式：使用额外的I/O管脚和芯片内部电路相连，该连线通过划片槽，这样芯片划断后就不能通过简单控制该管脚进入测试态。由于FIB修补技术的出现，这个手段已经过时。图2-4(b)是最有潜力的替代方案：在划片槽和邻近的芯片中设计部分控制电路，从而得到不可逆的测试态控制手段。

由于测试态提供了快速、全面访问存储器的机制，因此在晶圆测试完成后，需要将测试态永久关闭。然而，攻击者可以使用两个微型探针将熔断的熔丝桥接起来，重新启用测试程序，读出存储器内容。

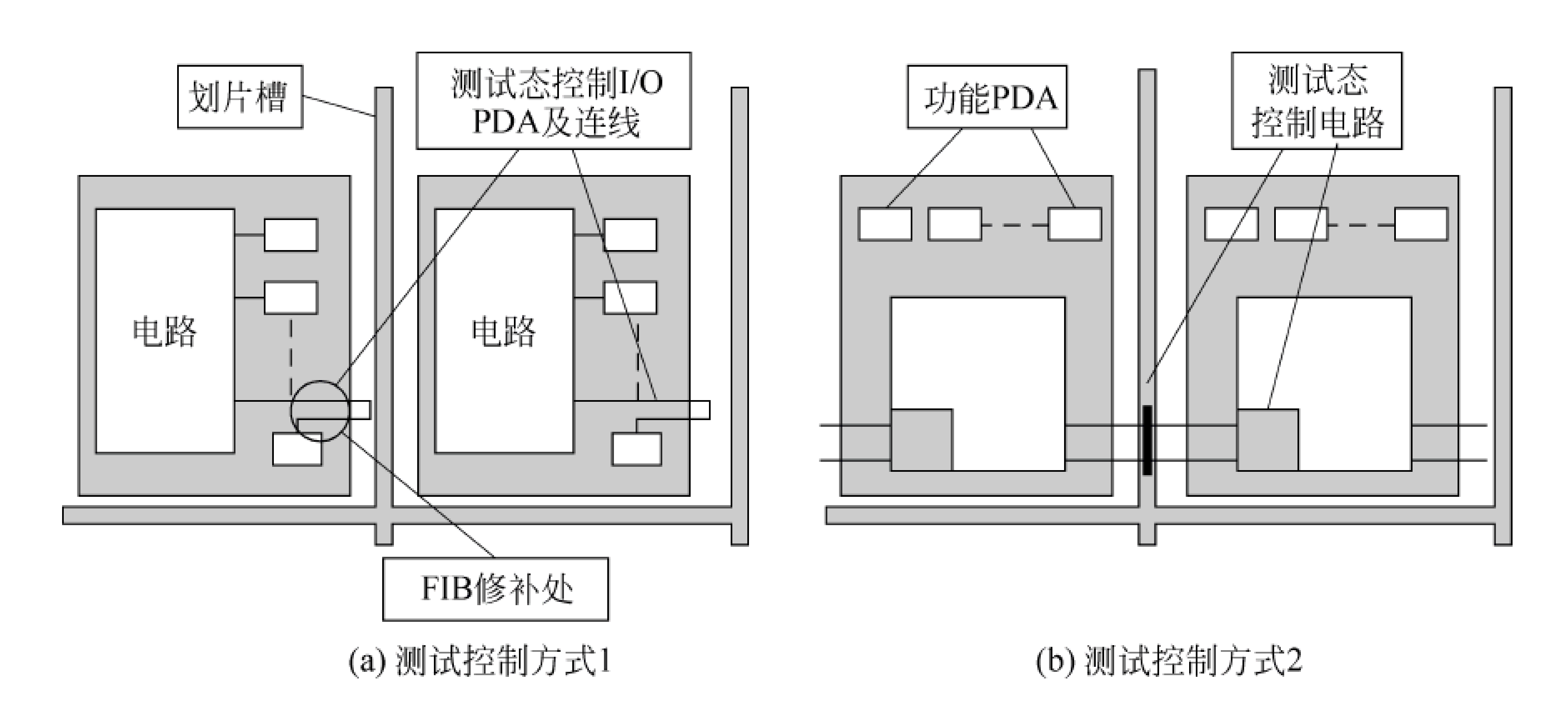


图 2-4 测试态控制方式

3. 读取 ROM

ROM 里面一般不存储任何和密钥相关的信息，但是它包含 I/O 信息、访问控制信息以及加密程序，这些信息在非入侵式攻击中都很有用。

光学重建技术可以直接用来读取 ROM。ROM 的位模式存储在扩散层，这使数据在芯片的表面不会留下任何光学迹象。有的 ROM 存储位不是活跃的，而需要通过修改晶体管电压阈值来将其激活。在这种情况下，攻击还要用到使数据位可见的选择性染色技术。

4. 改写 EEPROM

由于智能卡所有的密钥都存储在 EEPROM 中，而 EEPROM 的写操作会受到非正常电压和温度的影响，所以可以通过升高或者降低芯片供电电压来捕获 EEPROM 中存储的信息。简而言之，EEPROM 的最大问题就是非正常的电压和温度会影响它的写操作。

一个有名的攻击例子是针对 PIC16C84 芯片的：攻击者通过升高供电电压，使得该芯片在内容没有被擦除的情况下清除了安全控制位。另一个攻击是在 DS5000 安全芯片上实现的：一个短时的低电压脉冲使得芯片安全锁被解除了。

低电压还可以促使一些和 EEPROM 无关的攻击成功，比如，在电压被降低的情况下，用于产生密钥的随机数发生器产生的输出几乎全是 1。

5. RAM Remanence 攻击

随机存储器(RAM)，在断电之后还能在短时间内保存数据，尤其是当数据在 RAM 中存在的时间比较长的情况下，必然留下痕迹。这是进行 RAM Remanence 攻击的基础，攻击者可以利用这一特性恢复密钥。

2.1.3 获取密钥

攻击者进行针对智能卡存储器的攻击，主要目的是获取其中的敏感信息，尤其是密钥。

1. 使用 ROM 覆盖技术获取密钥

ROM 中的单个位都可以使用显微镜激光刀具进行覆盖，数据位被覆盖之后有时能使编码改变，甚至导致密钥的暴露。

DES 是一个很好的例子。DES 算法是众所周知的，攻击者可以找到某一位或者某几个

位，通过改变它们，容易萃取出密钥。萃取的具体细节取决于 DES 的执行过程，但是攻击者可以执行 jump 指令，将运算减少为一轮或者两轮。

另外，DES 的 S 盒在 ROM 中的位置可以被确定，可以对其进行覆盖，使 S 盒变成线性函数，这样，攻击者只需要使用线性函数破解技术就可以成功地获取密钥。

2. 使用 EEPROM 覆盖技术获取密钥

如果攻击者能修改 EEPROM 的内容，Key 也可能被恢复。在学术上很受关注的是和 DES 相关的一个例子。该攻击方法是利用奇偶校验实现的。假设攻击者知道 DES 密钥在 EEPROM 中的位置，并且能对 EEPROM 的任意位置进行修改，但是不能直接读出 EEPROM 数据。攻击的过程简述如下：

将 EEPROM 中包含目标密钥的第一个位置为 1（或者置为 0，这对攻击没有影响）。运行该芯片，如果它依然工作，那么密钥的第一位就是 1；反之，如果得到的返回是“密钥奇偶错误”，那么该位就是 0。之后再对第二位第三位等进行类似操作，直到得到密钥值。

3. 使用门摧毁技术获取密钥

在 DES 加密过程中，如果攻击者能破坏寄存器的一个门，使得此处在整个加密过程中保持一个常量的状态，那么密钥就能被恢复。

硬件实现的 DES 是典型的单轮计算，而且寄存器会存储 k 轮的计算结果并且将其返回，作为 $k+1$ 轮的输入。如果寄存器的最低位被卡住，相当于本轮计算输出的最低位为 0。通过比较左边和右边的低六位，部分密钥位就能被恢复出来。提供 10 组这种被损坏的密文，该轮密钥就可以根据微分密码分析技术推理出来。足够多的轮密钥可以大大简化 DES 密钥的搜寻工作。

这是第一个在完全不知道明文的情况下对 DES 进行攻击的方法。

4. 通过单总线位探测恢复密钥

如果能在加密过程中于本地观察一部分 RAM 存储的数据或者地址总线位，攻击者能很容易地恢复密钥的相关信息。这种攻击方法适用于非对称密码算法，如 RSA，也适用于对称密码算法，如 DES、RC5。

1）对非对称密码算法的总线探测

Helena 等人提出了本攻击方法。本方法完整地恢复了典型的指数-乘法运算的指数，这为破解 RSA、DSA 等算法提供了工具。

该模型假设攻击者能在内部指数-乘法运算执行后，获得某个累加器的值。这就意味着，攻击者能在累加器被平方或者平方-乘积之后就获得数据位的值。

Helena 展示了在知道 m 和 n 以及求幂函数情况下，如何通过探测 $m^d \bmod n$ 推出私钥 d。静态统计学分析显示，需要猜测的数字不会呈指数级增长，这种攻击方法是切实可行的。

2）对 DES 的总线探测

对于 DES 运算，只要能在每一轮都获得一个数据位的数据，就算是被动攻击者，也能通过这些数据恢复出密钥。

攻击者在本地观察 DES 运算中某一特定位的值，假设他有能力识别 R 和 L 数据寄存器各一个。不管是轮运算中的 R 寄存器还是 L 寄存器，知道它们中的任何一个位，都足以攻破第一个或者最后一个轮运算子密钥。使用 6 组不同的密文，可以恢复出 6 位密钥数据。

使用同样的攻击方法，可以得出第一个轮计算中的 6 位密钥。所以，总共有 12 位的密钥能通过此攻击方法获得。密钥剩下的 44 位数据可以通过穷举搜索得到。至此，就恢复出了所有的 56 位密钥。

3）对 RC5 算法的总线探测

假设攻击者通过探测，从 L 和 R 寄存器中成功获得计算密钥 b 的一些中间值。从寄存器中获得的中间值可以帮助攻击者推算出完整的扩展密钥，进而恢复出原始密钥值。这种攻击的复杂性非常低，甚至比穷举 32 位数据还要简单。

2.2 旁路攻击

什么是旁路攻击？可以想象这样一个场景：正常的通信过程中，各个终端都在进行数据交换，我们可以把每一个终端看作是一个运算设备，这些运算设备在工作的同时总会泄露出一些附带信息(包括功耗、错误信息、电磁辐射和执行时间等)，攻击者利用采集设备检测和搜集所有与保密数据相关的泄露信息，哪怕只是得到只言片语也能经过大量信息样本推算出想要得到的数据，而这一切各个终端却全然无知。

图 2-5 形象地展示了旁路攻击中最常见的一种情况，代表了旁路攻击的一种特殊性，由此可以推广到普遍性的结论，即任何密码系统在正常工作时都会或多或少地泄露与密码相关的旁路信息，攻击者在“系统不知不觉”的状态下搜集大量泄露信息进行密码破解的过程就称为旁路攻击。从普遍性结论可以引申出智能卡的旁路攻击推论，那就是每个芯片工作时都会泄露附带的信息，这些信息都有可能被攻击者利用进行密码破解。对于智能卡，通常是通过功耗和电磁辐射来提取信息的，具体包括简单功耗分析(SPA)、差分功耗分析(DPA)、高阶差分功耗分析(HO-DPA)、电磁分析(EMA)。

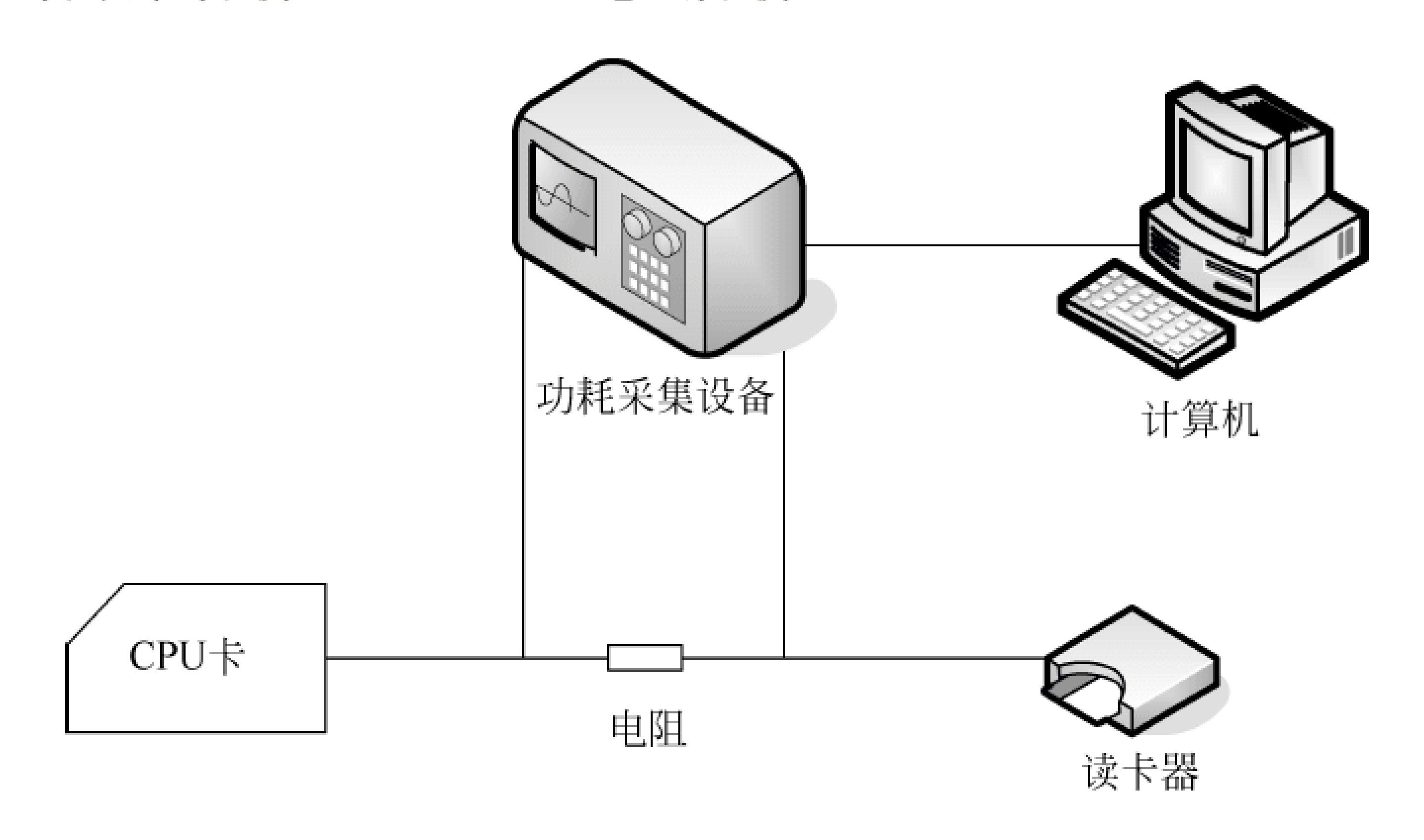


图 2-5 功耗采集示意图

旁路攻击范畴中，功耗攻击是目前攻击者常用的攻击方法，这种攻击手段具有攻击周期短和攻击代价低的特点。攻击者关注的是密钥参与运算时无法避免而泄露出来的功耗信息。对于 CPU 卡而言，攻击者只要在其 VCC 或 GND 上串联一个小小的电阻就能轻易获得芯片的功耗信息。

2.2.1 简单功耗分析

简单功耗分析(simple power analysis,SPA)攻击是指从密码芯片运算的功耗波形上能够直接找出一切密钥信息的攻击手段。在芯片中,密钥的存在形式都是 0 和 1 的二值代码,根据密钥某一位数值的不同,密码运算的步骤也大不一样,这就决定了数据在电路里经过的路径也大相径庭,造成的结果是密钥位为 0 时运算的功耗和密钥位为 1 时运算的功耗存在差异,攻击者就是通过这样的差异分析出密钥位的值,最后将其按一定的规律组合起来就得到了完整的密钥。

针对 CPU 卡,SPA 攻击的主要任务就是采集密码协处理器工作时的功耗,当一次完整的加密计算完成之际,攻击者也同时得到了此次加密的完整功耗波形,若密钥位 0 和 1 运算的差异越大,则体现在功耗上的特征就越明显,攻击者甚至无须采集多次就能得到密钥,密钥位 0 和 1 的功耗差异的示意图如图 2-6 所示。

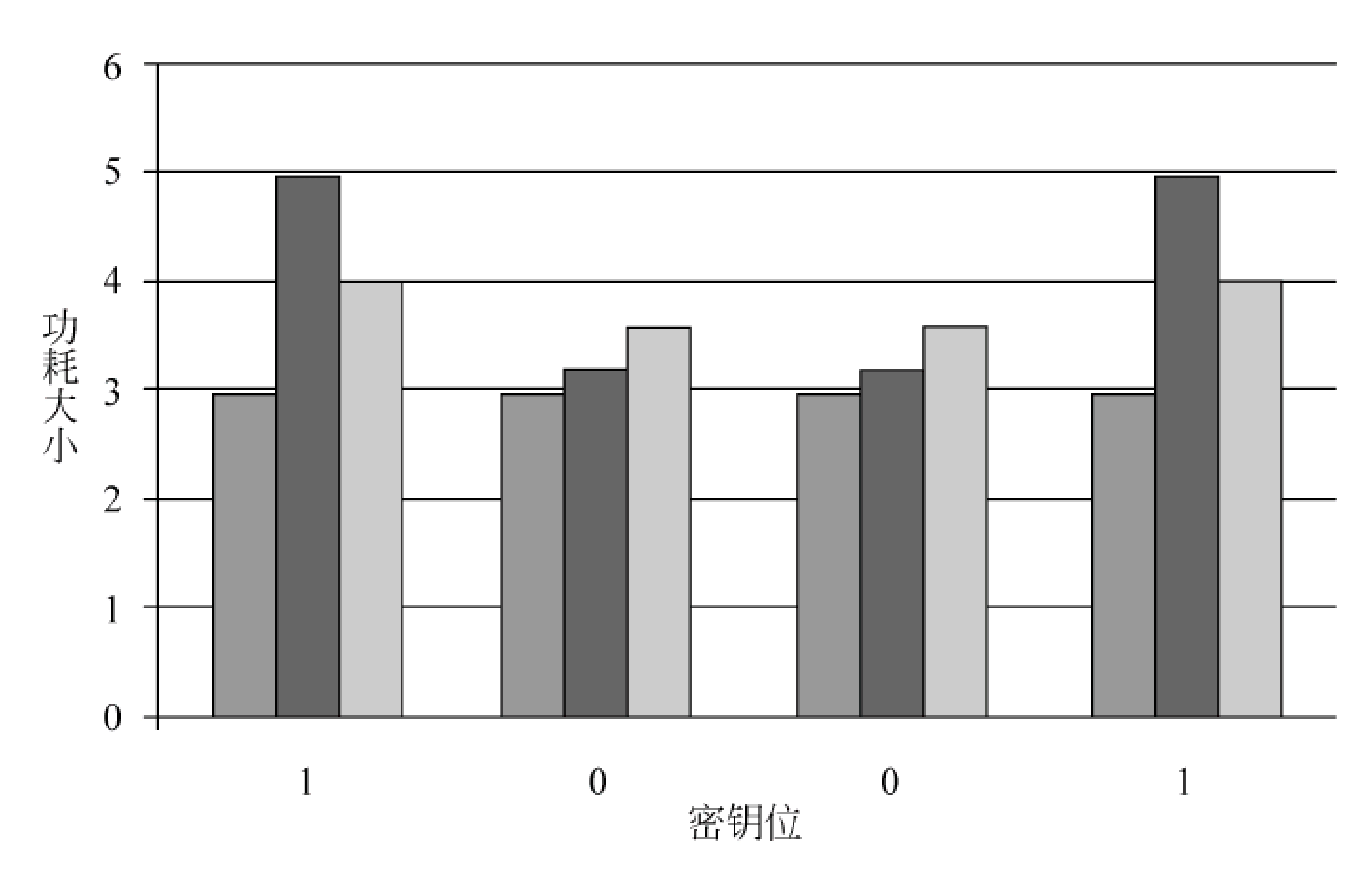


图 2-6 SPA 功耗差异示意图

在图 2-6 中,横坐标表示密钥位的值,纵坐标表示功耗大小,该图表达了 SPA 攻击的根据,实际中的功耗特征可能不会像这样十分明显,但是差异的存在是客观的。因此 SPA 可以看作是很直接的攻击方法,攻击消耗的代价几乎可以忽略不计,其威胁性可见非同一般。

然而实施 SPA 需要有一些附加的条件约束,首先对 SPA 攻击者有一定的要求,攻击者必须知道芯片中密码的具体算法;另外就是密钥位 0 和 1 的运算功耗必须有差异,否则 SPA 攻击将无效,可见一些平衡功耗的做法对防御 SPA 攻击是相当有效的。

2.2.2 差分功耗分析

差分功耗分析(differential power analysis,DPA)攻击是通过用示波镜检测电子器件的能量消耗来获知其行为。攻击者只需知道算法的明文(输入)或密文(输出),通过分析和比较一系列的能量轨迹就可重现加密密钥。

DPA 攻击的主要任务在于找出密码运算时功耗和密钥位的相关性,而不是直接得到密钥位的值。在某些特殊的条件下,SPA 攻击无法实施,那么攻击者就会通过 DPA 的手段破解密钥。虽然 DPA 的依据也是密钥位数值不同引起的功耗差异,但是这种差异不是直观

的。换而言之，无法像图 2-6 那样能直接从功耗波形上看出 0 和 1。DPA 的复杂程度比 SPA 高得多，攻击者的目标不再是密钥位自身直接表现出的功耗差异，而是从密钥参与计算后的结果中分析功耗差异和密钥的相关性，攻击的主要任务是针对密钥获取足够多的功耗曲线，综合起来做统计分析。

就 DPA 本身而言，复杂程度和具体选用的方案也有关，比如在公钥密码体系(PKI)中，DPA 攻击常用的方案有多指数单数据攻击(MESD)和单指数多数据攻击(SEMD)，它们的共同点是先假设所要攻击的密钥位，接着通过统计分析验证假设正确与否。

1. MESD 方案

MESD 的操作方法是不更换明文，不断地变换密钥，逐位破解。具体操作流程是：首先进行一次加密计算，得到一条功耗曲线；接着从密钥的最高位开始，假设要攻击的位为 1，其他低位为 0，用假设的密钥进行加密计算，又得到一条功耗曲线；然后将这两条功耗曲线相减，就得到了一条差分功耗曲线，在差分功耗曲线中所假设密钥位对应的时间点上，若幅值接近 0 则说明假设正确，该密钥位为 1，否则说明假设错误，该密钥位为 0；最后按照此方法由高到低破解密钥的每一位。

我们可以看这样一个例子，密钥的二进制表示为$(10100\cdots)_2$，下面用 MESD 方案破解该密钥的次高位。

步骤 1：加密已知明文 P，获取功耗曲线 C；

步骤 2：假设密钥的次高位为 1，其他低位全为 0，那么假设的密钥就是$(11000\cdots)_2$，用假设的密钥加密明文 P，得到功耗曲线 C1；

步骤 3：用 C 减去 C1，计算出差分功耗曲线。图 2-7 为差分功耗曲线的示意图。

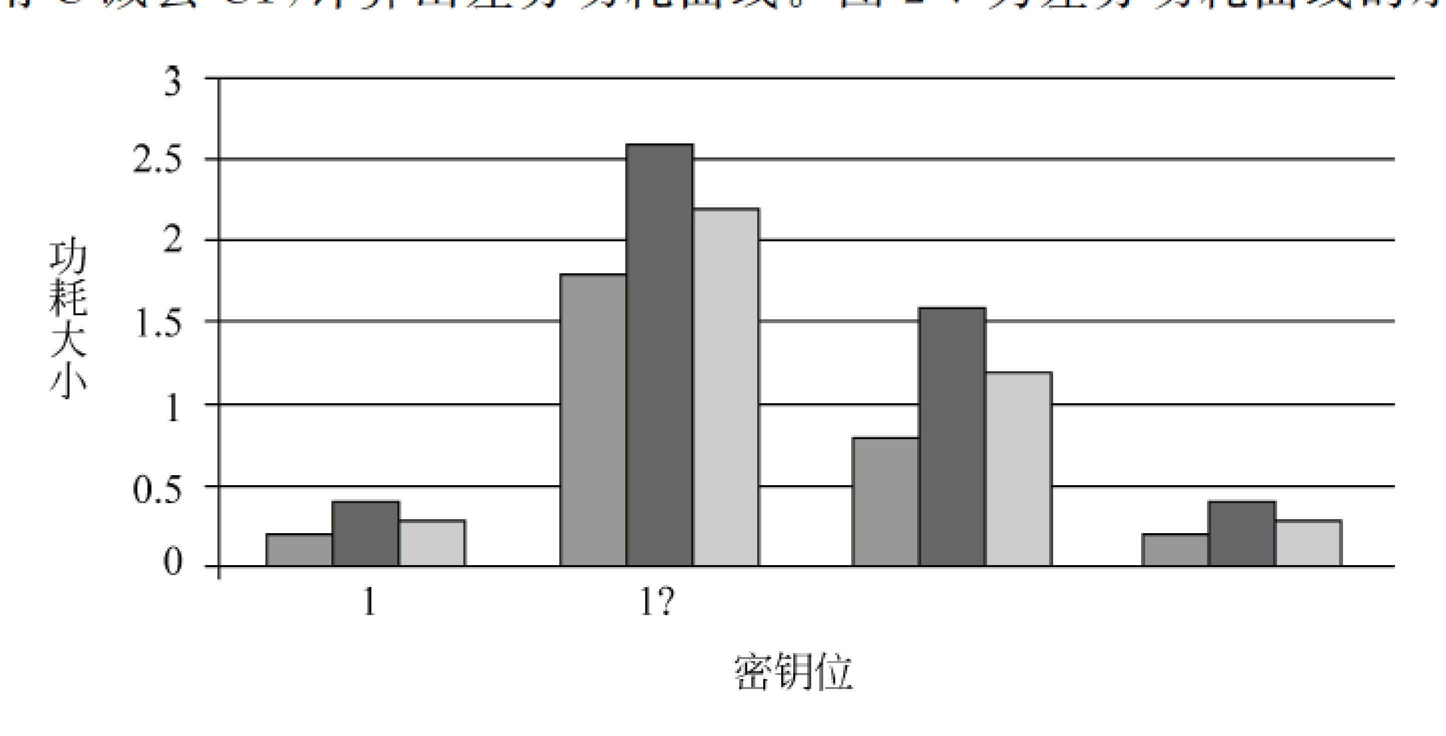


图 2-7 差分功耗曲线示意图

图 2-7 表示的是密钥次高位所对应时间点上的差分功耗波形，该波形中的幅值并不接近 0，验证了之前的假设是错误的，因此原密钥的次高位应为 0。幅值不为 0 的理由是假设的密钥位和真实密钥位不一致，所以计算的过程不一样，导致在同一时间点上的功耗不同。以上举例说明了 MESD 方案的 DPA 攻击，其破解步骤算不上很复杂，但是 MESD 也有自己的附加条件限制，即攻击者必须有密钥擦写的权限，如果没有该权限就不能更换密钥，所以也没有办法进行 MESD 攻击。

2. SEMD 方案

SEMD 的操作方法是不更换密钥，不断地变换明文，逐位破解。SEMD 的破解过程比较复杂，所消耗的时间代价也是很巨大的。实施 SEMD 必须要建立一个和芯片相同算法的

加密模型，这个模型可以使用软件实现，也可以使用硬件实现，而且要把加密算法分割为多轮进行破解，每次破解一轮，最后将破解的结果组合起来就得到了密钥。

首先从密钥的最高位开始假设要攻击的密钥位为0，针对每一位密钥的攻击都是破解该密钥位对应的那一轮加密运算；接着在加密模型中对明文进行这一轮加密计算，查看加密结果中某一个固定位的值为0还是1；然后在芯片中对同一明文进行加密，获取功耗曲线，若加密模型中，结果的某一个固定位的值为0，就将这条功耗曲线放在0集合中，否则将其放入1集合中；接着按照上述过程把所有明文加密的功耗曲线分为两组，并计算每一组的平均功耗曲线；随后把这两个分组的平均功耗曲线相减得到差分功耗曲线，在差分曲线中所假设密钥位对应的时间点上，若出现尖峰则说明假设正确，该密钥位为0，否则说明假设错误，该密钥位为1；最后按照此方法由高到低破解密钥的每一位。

我们这里举一个例子描述SEMD方案的实施过程，密钥的二进制表示为$(101001000\cdots)_2$，破解密钥的次高位。

步骤1：建立加密模型，准备n组明文；

步骤2：假设密钥的次高位为0，在加密模型中以$(10)_2$为密钥对其中一组明文进行加密，查看加密结果的最低位是0还是1；

步骤3：在芯片中加密上一步中使用的明文，获取功耗曲线，根据步骤2中的结果将其分到0集合或者1集合；

步骤4：更换另一组明文，从步骤2开始执行，直到加密完所有明文；

步骤5：分别计算0集合和1集合的平均功耗曲线；

步骤6：将两个集合的平均功耗曲线相减得到差分功耗曲线，在密钥次高位对应的时间点上会出现尖峰，说明之前的假设是正确的，原密钥的次高位应该就是0。因为假设的密钥位数值和原密钥位一致，所以加密模型的结果也是对的，这就保证了后续分组的正确性。结果中0与1的计算过程有功耗差异，因此在差分功耗曲线相应的位置上会出现尖峰。

SEMD方案虽然复杂，但是没有任何的附加条件，实施不受限制，任何人、任何时间以任何地点都能进行攻击，成为CPU卡的最大威胁。

2.2.3 高阶差分功耗分析

DPA研究的是秘密数据和功耗曲线上一个点的关联，而高阶差分功耗分析(high order differential power analysis，HO-DPA)研究的是秘密数据和功耗曲线上几个点的关联，HO-DPA攻击实施的难度较高，但攻击能力增强。

HO-DPA的主要思想是关联几个值，以便得到重要数据的功耗。攻击需要的数据样本数量和计算量远远大于DPA，这导致HO-DPA攻击难度增加，限制了其应用。HO-DPA在实施攻击过程中，攻击对象是依赖少于32位密钥的数据。如果依赖高于32位密钥的数据，对其实施攻击代价太大，失去了旁路攻击的意义。

HO-DPA中最常见的是二阶DPA。

2.2.4 差分电磁分析

随着近年来对电磁信息泄露的研究，人们发现电磁辐射泄露的信息不仅丰富，而且在攻击时无须分解设备和改动电路，具有很强的实用性。

微处理器内部的各种部件,特别是与数据操作有关的控制器、运算器和总线等部件,在数据处理过程中产生的电磁辐射会带来敏感信息的泄露。这些部件在时钟控制下,其状态变化取决于运算、操作对象以及运算结果,这种状态变化称为部件的相关状态。相关状态引起的电流变化会产生电磁辐射,而这些辐射信号时刻反映出部件的相关状态,使操作数据和电磁辐射之间存在着相关性,通过对辐射信号时、频域特性的分析可以获得操作对象以及运算结果等信息。在加密系统中,所有的运算操作都将转化为对逻辑 0 和逻辑 1 的操作,电路中对逻辑 0 和逻辑 1 的处理由于不同的电流变化会产生不同强度、不同频谱特征的电磁辐射,因此,使用电场或磁场探头测量密码系统产生的电磁辐射信号,利用数据与信号的相关性,就可获得加密系统的密钥。

差分电磁分析(differential electromagnetism analysis,DEMA)的基本原理是由于在加密过程中芯片产生电磁辐射,而辐射能量的大小随处理数据的不同会有微小的变化,对这种变化采用差分统计的方法可以确定所处理数据是 0 还是 1,从而有可能猜出加密算法中所使用的密钥。DEMA 攻击的实验平台如图 2-8 所示,示波器利用探头来获取智能卡密码系统的电磁辐射,然后交由计算机进行分析。

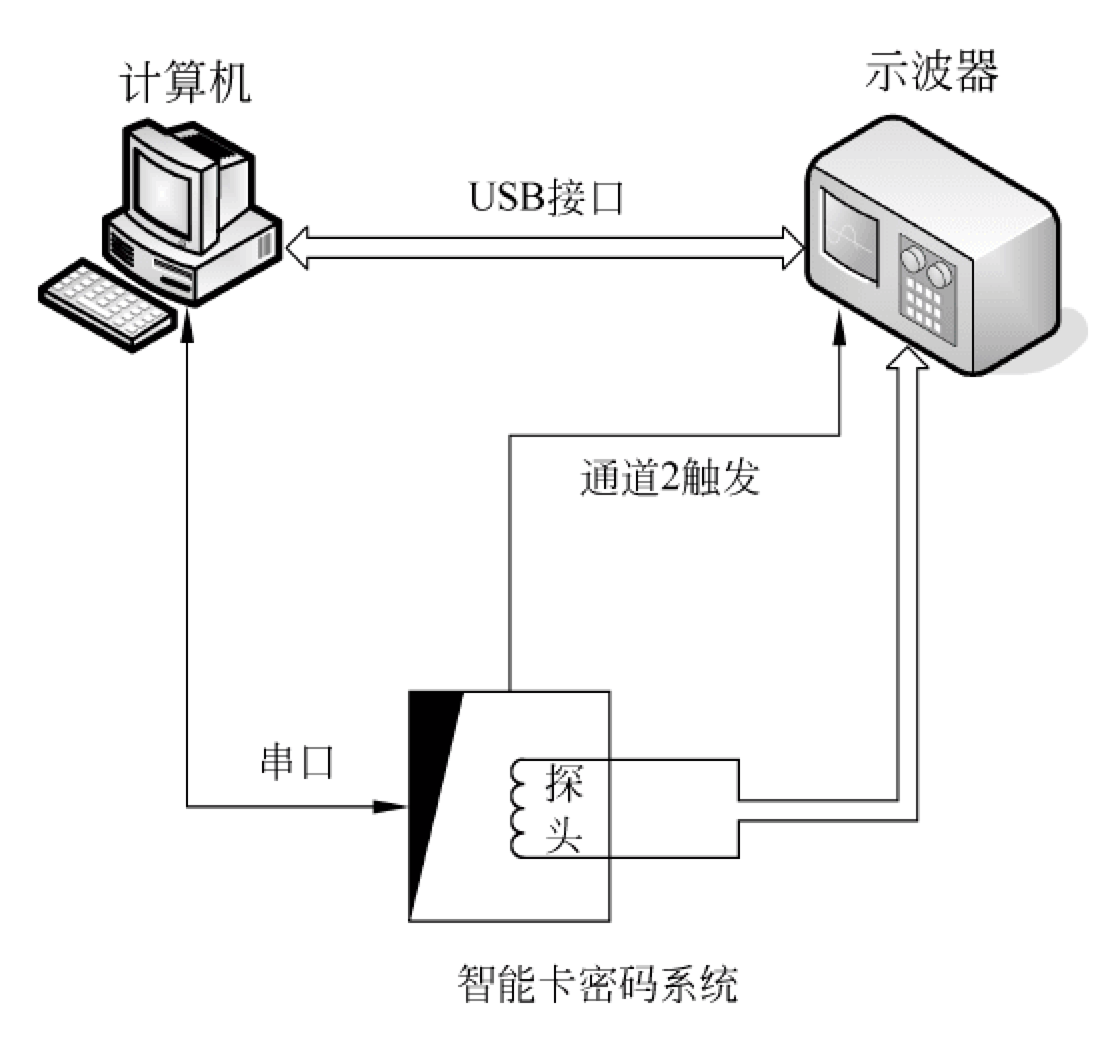


图 2-8　DEMA 平台

差分电磁分析算法是基于假设检验理论的一种统计方法。攻击者首先建立一个关于被测数据的假设 D 函数,然后由大量的实际泄露样本计算检验统计量,从而判断假设 D 是否正确。在对 DES 算法进行差分电磁分析时,首先选择在密码操作过程中与密码相关的中间变量 temp 作为 D 函数,该函数定义规则为 $F(\text{temp},K_b)=\text{CT}$,其中 CT 为密文,$K_b$ 为 b 位子密钥,$F()$是一个 DES 加密算法中的某一确定函数,那么得到等式 $D(\text{CT},K_b)=\text{temp}$。

如果 temp 只有 1 位时,可将 $D(\text{CT},K_b)\to\{0,1\}$作为一个判别函数。它的意义在于 K_b 猜测的正确与否,同 temp 的正确分组密切相关,也就与计算 temp 产生的电磁辐射能量密切相关。因此,DEMA 攻击算法如下:

步骤 1　首先进行 M 次加密运算,经采样获取 M 条电磁辐射信号曲线。

(1) 随机产生 M 个明文 PT_i,$1\leqslant i\leqslant M$;

(2) 对每一次加密过程进行采样，得到离散化电磁辐射信号曲线数组 $S_i[j]$，其中 $1\leqslant j\leqslant N$，j 表示采样的时间序列；

(3) 同时得到对明文 PT_i 加密产生的密文 CT_i，$1\leqslant i\leqslant M$。

步骤 2 令 $K_b=0$，将电磁辐射曲线数组 $S_i[j]$根据下列条件分成两个子集合。

$$S_0 = \{ S_i[j] \mid D(\mathrm{CT}, K_b) = 0, (1 \leqslant i \leqslant M) \}$$

$$S_1 = \{ S_i[j] \mid D(\mathrm{CT}, K_b) = 1, (1 \leqslant i \leqslant M) \}$$

步骤 3 计算集合 S_0 与 S_1 的平均值 $A_0[j]$和 $A_1[j]$。

$$A_0[j] = \frac{1}{|S_0|} \sum_{S_i[j] \in S_0} S_i[j]$$

$$A_1[j] = \frac{1}{|S_1|} \sum_{S_i[j] \in S_1} S_i[j]$$

并得到电磁辐射信号的差分组 $T[j]=A_0[j]-A_1[j]$。

步骤 4 如果 $T[j]$在某个位置出现明显的尖峰，则表示分组正确，即 K_b 猜测正确，执行步骤 5。否则，修改密钥，令 $K_b=K_b+1(K_b\leqslant 2^b-1)$，转步骤 2，根据新的 K_b 再次分组，循环运行直到 K_b 猜测正确。

步骤 5 按上述方法获取所有子密钥块以后，可得到整个密钥。

2.3 半入侵式攻击

智能卡控制器通常采用硅片制成。而硅片的电性能会随着不同的环境参数而改变，例如，硅片的电性能将随着不同的电压、温度、光、电离辐射以及周围电磁场的变化而改变。攻击者通过改变这些环境参数，可以引入一些错误的行为，比如向智能卡控制器的程序流引入错误等。通常，这些错误的行为会让芯片做出错误的决定，例如接收错误的输入鉴权码，允许访问存储器中的保密数据。

目前，这种通过引入错误造成智能卡功能混乱的方法已经演变成一种攻击方法，被称为误感应攻击，也叫做半入侵式攻击。

半入侵式攻击主要包括差分错误分析攻击、能量短脉冲干扰攻击、时间分析攻击等，下面将分别进行介绍。

2.3.1 差分错误分析攻击

差分错误分析(differential fault analysis，DFA)是一种被广泛研究并使用的智能卡密钥攻击技术。攻击者使用破坏性或是非破坏性的技术，使密码相关的计算过程发生错误，并产生错误的计算结果，之后将这个错误的结果和原本应该产生的正确结果作比较，从两个结果的差异中，试着找出引起的错误和最后产生的结果之间的关系，进一步利用这个关系找出运算过程中所使用的密钥。

为了实施 DFA 攻击，攻击者需要进行以下步骤。

1. 了解想要攻击的密码运算，找出能引入错误的可能性

首先，一个重要的概念是 DFA 攻击的并不是 RSA 或 DES 这些密码运算的理论本身，而是攻击使用这些运算的方法。比如 RSA，它是一个存在已久、经过时间考验的密码运算

方法，在数学上是一个无法有效破解的困难问题。但是，当 RSA 被放在一个平台上（例如智能卡）执行时，RSA 运算的安全性就需要依靠不同层级、硬件、软件的合作来确保。以下就从四个不同方面进行分析。

1）密码运算执行方法

我们再次以 RSA 为例子来做说明。

RSA 运算在硬件上最常用的执行方法有两种：

（1）直接运算。

（2）使用中国剩余定理（Chinese remainder theorem，CRT）方法运算。

RSA 的加密方法是将要加密的信息和密钥做一个指数运算。直接运算是将信息的每一个位元，依照密钥的位元，做硬件的二次方和相乘运算；而 CRT 方法，则是先将信息和密钥分成原来的一半长度，再用这些长度比较短的信息和密钥来做硬件的二次方和相乘运算，最后把所得到的结果结合起来得到完整的运算结果。

在同样的硬件条件下，使用 CRT 能够将运算的时间加快约四倍，这使得 CRT 运算方法获得许多智能卡开发者的青睐。不过对于 DFA 来说，如果能够成功引入一个错误，整个密钥就能够被获取。由此可见，从最基本的运算执行方法的选择上，就存在着 DFA 的威胁。

2）密码运算编程

大部分的情况，RSA 是以软件形式存在于智能卡中的，这与 DES 不同。如果 RSA 的程序没有用安全的编程方法来编写，程序中很有可能会存在弱点，使得攻击者可以找到方法跳过运算中的一些步骤。同时，许多的攻击能够用软件来做预防和抵抗，大部分的智能卡芯片制造商都会建议顾客购买他们提供的 Crypto-Library，Crypto-Library 里面有制造商配合自己的硬件平台编写的安全密码运算方法。这个硬件平台和 Crypto-Library 的搭配，往往都已经通过 Common Criteria 高安全等级的检验，能够抵抗大部分已知的攻击，保障了使用芯片的安全。

3）硬件平台

对智能卡来说，整个产品的安全性，需要硬件和软件的共同配合和互补。就算在软件或是运算执行方法上存在着弱点，如果它们是搭配安全性极高的硬件平台，也能保障智能卡的使用安全。硬件平台能够使用攻击感应器等元件来察觉攻击。如果智能卡一旦察觉到攻击，就马上停止所有操作，那么 DFA 能够成功的几率就很小，或是就算成功也不再有用处。当然，如此一来智能卡可能对环境太敏感，会造成使用上的不便。然而由于目前最有效的错误引入方法都是物理性的攻击，因此硬件平台的安全性，是绝对事关重大的。

4）应用程序

目前有一些新的 DFA 攻击，专注在智能卡的应用程序的弱点上。当然，如前面强调的，一个产品的安全性，建立在不同层级，以及各个层级之间相互的合作之上。最近对于应用程序的攻击之一，是原本不会泄露的 RSA 运算的输入值，被发现能够在应用程序的另外一个步骤中被读出来，该值是在卡片内部由随机数产生器产生出来的。攻击者本来只能知道运算的结果，但是因为这个现象，现在能够知道 RSA 运算的输入值。攻击者能够用同样的输入值，经过错误的引入，找出引入错误和运算结果的关系，实施 DFA 攻击。

2. 找出引入错误的方法

在分析过 DFA 攻击在各个层级的可能性之后，攻击者便需要找出能够引入错误的方

法。引入错误基本上是在要攻击对象的平常功能上，造成扰动。下面介绍两种引入错误的方法。

1）电力扰动

当智能卡进行密码相关的运算时，攻击者在智能卡的一般电力供应入口上，提供短暂的超出一般电力范围的电压刺激，然后观察智能卡是否会因此产生错误的运算结果。这个攻击并不需要打开卡片的表面或对智能卡进行其他处理。

2）光束扰动

攻击者使用激光触发器，在芯片执行密码运算时，对芯片的表面发射短暂的、高能量的光束，借此在运算的过程中引入错误，如图 2-9 所示。

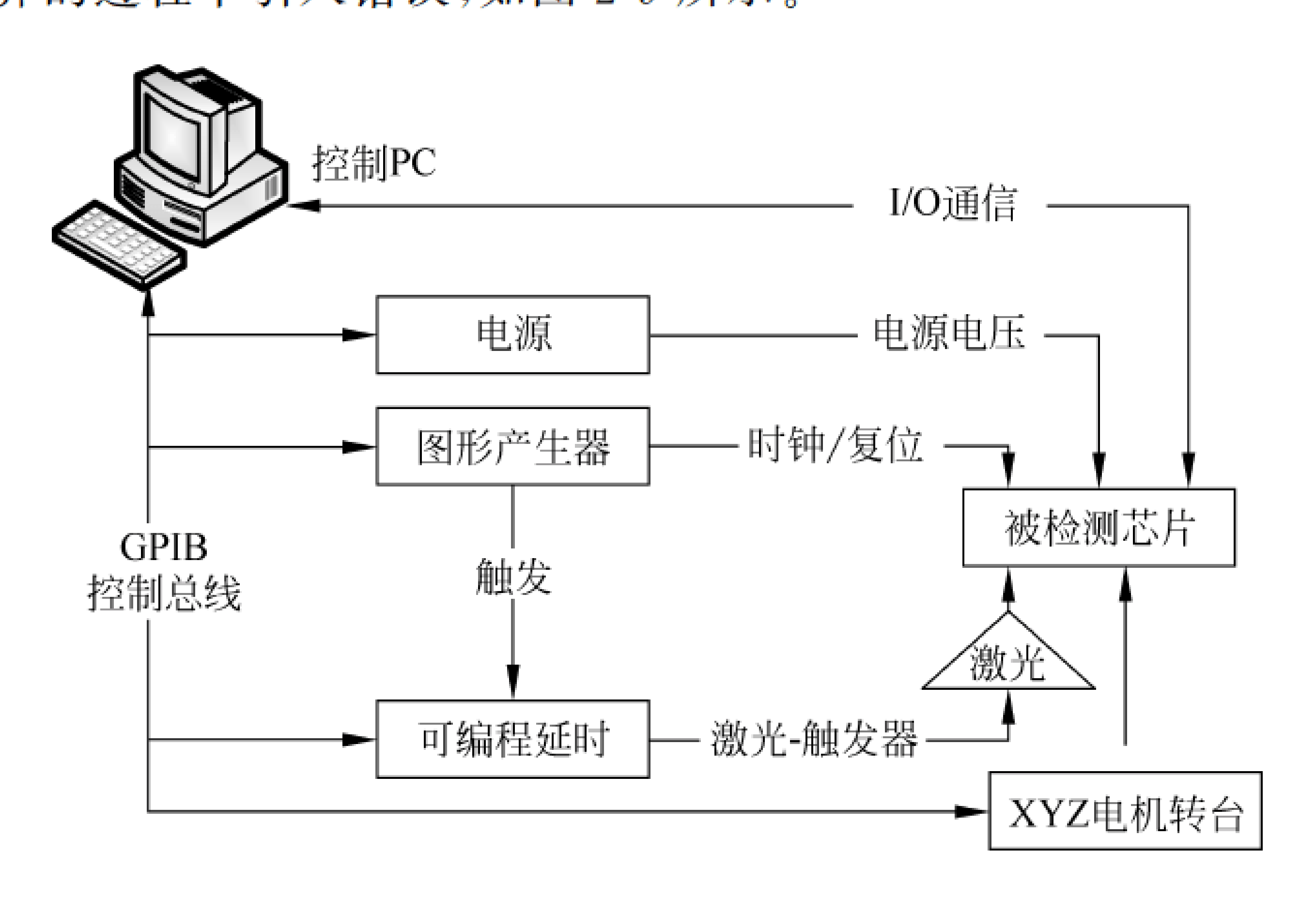


图 2-9 光束扰动示意图

该技术的基础是所有半导体产品都会被光束能量（光子）影响。光束扰动是在运算过程中引入错误的代表技术，也是目前最有效的错误引入方法。不过，它的复杂度非常高，如图 2-10 所示，攻击者需要对硬件、应用程序、电流分析和光学元件有深度的了解和丰富的经验。



图 2-10 光束扰动设备

3. 分析所产生的错误，并获取密钥

当一个错误被成功地引入（没有造成断电或智能卡的锁卡），并使得错误的运算结果产生，攻击者需要去分析所得到的错误运算结果。错误的运算结果是事先所预期的吗？密钥的确能够依照理论被判断出来吗？事实上，整个过程充满着很多的不确定因素。智能卡使用的芯片，随着时间的推移和技术的进步，运算速度变得越来越快，元件面积变得越来越小，防护措施越来越完善。很多时候虽然理论上攻击是可行的，但是事实上，要在运算过程中正确的时间点，在芯片正确的位置成功引入错误，仍然是一个不小的挑战。

DFA 是一个已经有一定历史，而且被详细研究的攻击方法。许多针对不同密码学算法（包括 RSA 和 DES）的 DFA 技术都已经被发表。DFA 能够在不同系统的不同层级被实施，正因为它强大的特质，它是智能卡硬件和软件产品设计者都需要去详细考虑并做防备的攻击之一。

2.3.2 能量短脉冲干扰攻击

短脉冲干扰（glitch attack）常用的第 1 种攻击方式：微处理器要求在稳定的电压下工作，能量供应的中断就好像突然冲击程序运行或复位电路。然而，一个短而巧妙的脉冲可以引起单步的程序错误而微处理器仍能继续执行程序。例如，CPU 读取存储单元的内容，三极管用一个阈值来检测存储单元的值，以确定所读的是逻辑 0 或 1。突然出现的能量短脉冲对存储值和逻辑值都会产生影响。如图 2-11 所示，与逻辑 0 对应的低电平在正常的操作状态下可能低于阈值电平，然而由于短脉冲的能量下压可能导致其高于阈值电平。许多加密算法都容易受这一类故障注入的影响。之后采用 DFA 技术，将正确的与错误的密码编码相比较，从而分析出密钥。

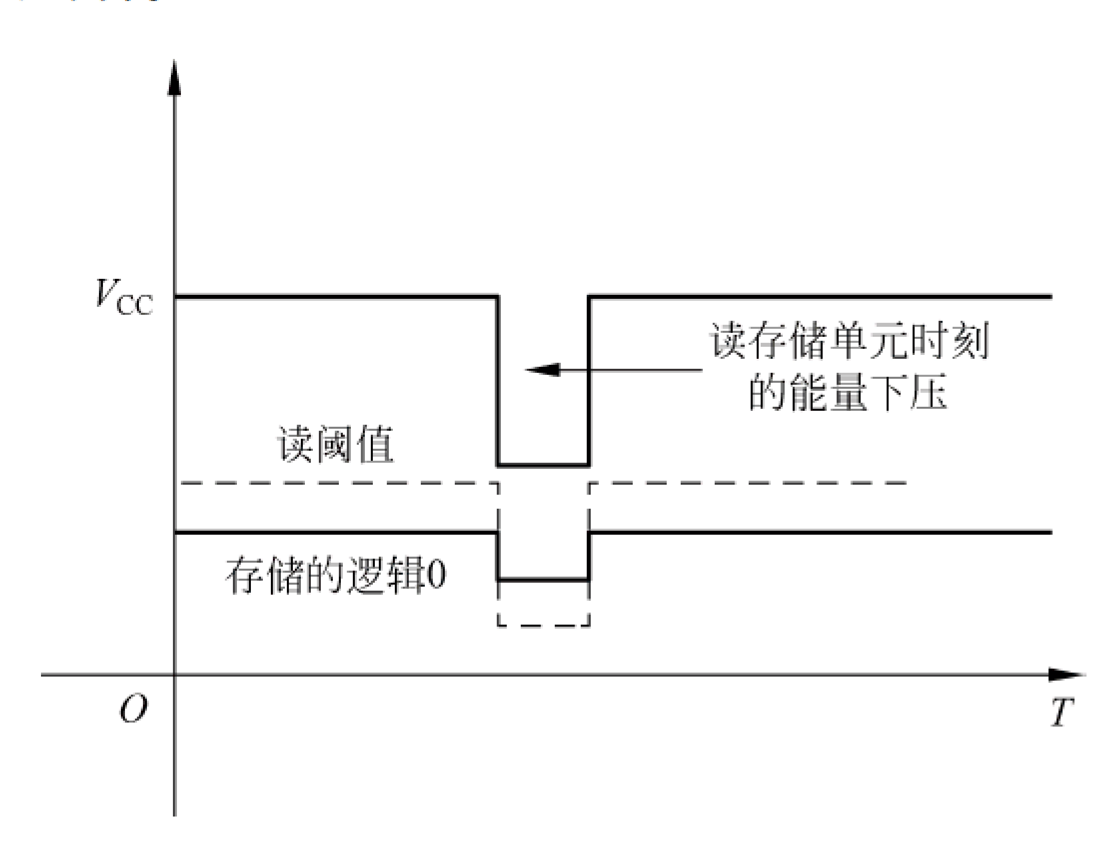


图 2-11 读存储器时能量短脉冲干扰

短脉冲干扰的第 2 种攻击方式是将 PIN 校验失败转为成功以欺骗处理器。更为严格的一种方式是在处理器正要将校验失败写入存储器时完全关闭电源，从而避免 PIN 校验失败计数器溢出。

短脉冲干扰的第 3 种应用是攻击发送限制计数器，从而导致整个存储器内容输出到串行接口。

2.3.3 时间分析攻击

时间分析攻击是基于对某一单元完成操作所需时间进行的测量,该信息能和密钥建立起某种联系。比如,通过对执行私钥操作所需时间进行精确的测量,攻击者可能会找到 DH (diffie-hellman)算法的指数、RSA 算法的因数等,进而攻破密码系统。如果密码系统的某个单元是脆弱的,时间分析攻击执行起来就会比较简单,而且通常只需要密文。

密码系统在处理不同输入时,往往只会有细微的时间差别,因为密码系统会通过优化算法,绕过不必要的操作,比如分支条件语句、内存缓存采样、随机执行的处理器指令说明等等。密码系统执行时间主要依赖于密钥和输入数据这两个要素。

2.4 通信链路攻击

读写机具和智能卡之间存在接触或者射频通信通道,攻击者可以选择在此通信链路中进行信号干扰、数据窃听等多种攻击,攻击模型参见图 2-12。这些攻击威胁着智能卡的可用性、机密性和完整性。

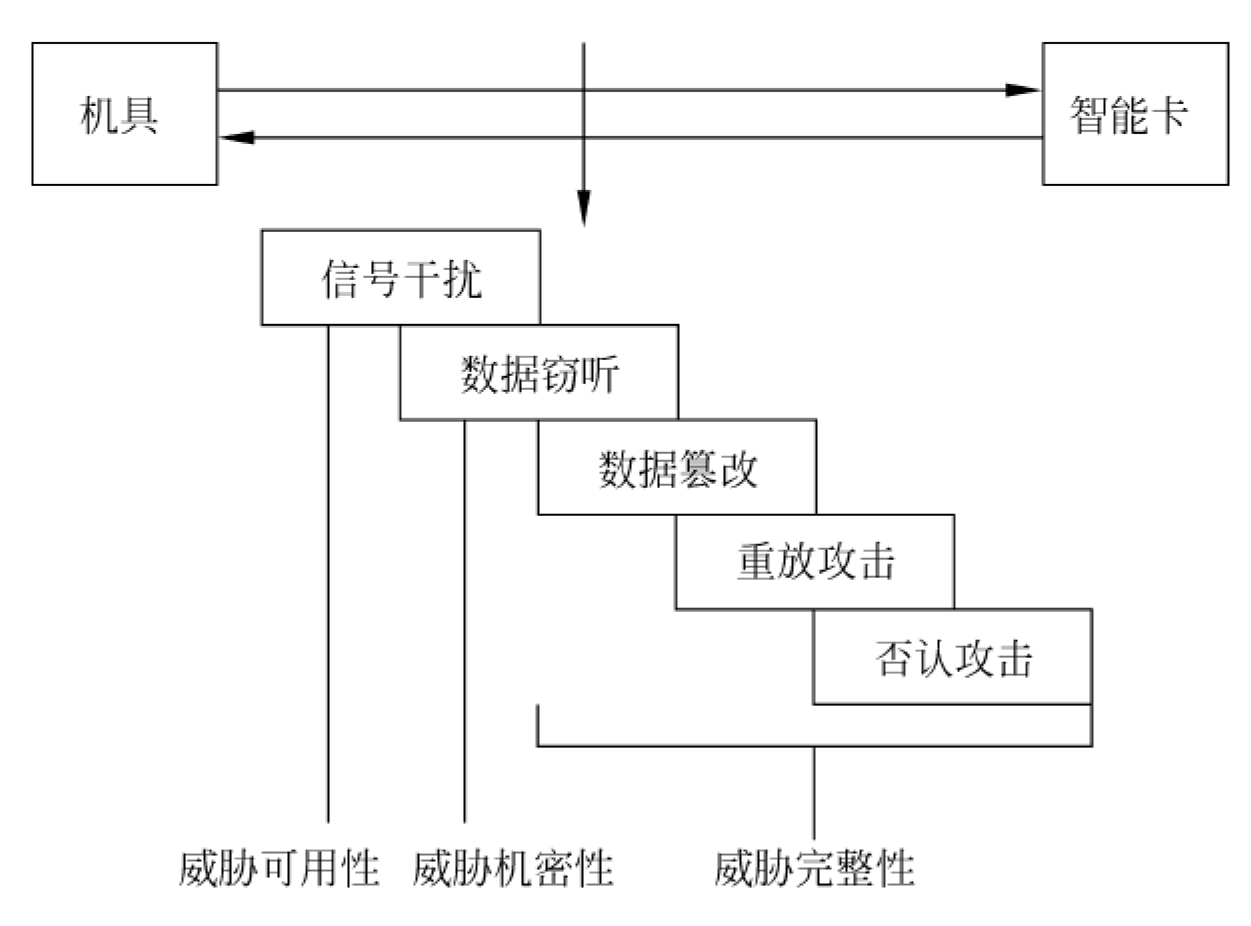


图 2-12 通信链路攻击模型图

2.4.1 信号干扰

信号干扰主要是针对非接触式 IC 卡的。攻击者可以利用其他设备发射和 IC 卡相同频率的信号,或者使用大功率的读写机具,干扰合法机具对 IC 卡的正常读写,以达到扰乱正常会话过程的目的。

2.4.2 数据窃听与篡改

常见的通信通道截获装置:在智能卡 I/O 接触面的顶部粘贴一个绝缘的哑触头,新的(哑的)触头和原来的 I/O 触头都接至一台计算机,经适当编程,这台计算机就可以截获智能卡和机具之间的通信数据,甚至删除或插入任何数据到机具和智能卡的通信中去。如果计算机运算速度足够快,机具和卡都不能检查出正常的和被操纵的通信之间的区别。显然,

用上述方法能影响机具和智能卡之间的会话过程。攻击者是否能由此方法获得利益取决于智能卡的应用领域。目前,已经有使用此类攻击进行诈骗的案例。

2.4.3 重放攻击

重放攻击就是攻击者发送一个目标智能卡(或机具)已经接收过的包,来达到欺骗智能卡(或机具)的目的。重放攻击主要用于身份认证过程。攻击者利用监听或者其他方式盗取认证凭据,之后再把它重新发给攻击目标。对通信数据进行加密可以有效地防止窃听,但却防止不了重放攻击。

根据消息的来源不同,重放攻击可以分为协议轮内攻击和协议轮外攻击。前者是指一个协议轮内的消息重放;后者是指一个协议不同轮次消息重放。

根据消息的去向不同,重放攻击可以分为偏转攻击和直接攻击。其中,偏转攻击改变消息的去向,可分为反射攻击和第三方攻击。

- 反射攻击:将消息返回给发送者;
- 第三方攻击:将消息发给协议和通信双方之外的任意一方。

而直接攻击是将消息发送给意定接收方。

在身份认证过程,可将一次性随机信息融于认证过程,则可预防重放攻击。

2.4.4 否认攻击

否认攻击是指交易双方中的一方在交易完成后否认其参与了此交易。这种威胁在电子商务中十分常见。

目前已经成为众多银行和主要电子商务企业首选安全认证解决方案的 USBKey 技术可以有效地防止否认攻击。每一个 USBKey 都具有硬件 PIN 码保护,PIN 码和硬件构成了用户使用 USBKey 的两个必要因素。用户只有同时取得了 USBKey 和对应的用户 PIN 码,才可以登录系统。即使用户的 PIN 码被泄漏,只要用户持有的 USBKey 不被盗取,合法用户的身份就不会被仿冒;如果用户的 USBKey 遗失,拾到者由于不知道用户 PIN 码,也无法仿冒合法用户的身份。因此经由某一个 USBKey 产生的交易记录是不能被否认的。

以上介绍了常见的智能卡通信链路攻击方法,针对这些攻击的安全措施将在本书第 5 章中详细描述。

2.5 COS 逻辑攻击

COS 在智能卡中的地位和作用很像在微机上使用的各种操作系统。在 COS 中,智能卡芯片内各种硬件与用户所要求的应用系统密切结合起来,实实在在地体现出智能卡的安全所在。

COS 逻辑攻击的主要方法是对处理器的通信接口进行分析,以便发现智能卡协议、密码算法及其实现过程中潜藏的逻辑缺陷,包括未使用的命令、不良参数、缓冲区溢出、文件存取漏洞、恶意攻击、通信协议和加密协议的设计与执行过程等。逻辑攻击者在软件的执行过程中插入窃听程序,利用这些缺陷诱骗卡片泄露机密数据或允许非期望的数据修改。

逻辑攻击主要包括智能卡 COS 软件木马攻击、协议攻击和安全体系攻击等。

2.5.1 木马攻击

COS 软件是嵌入式软件，通常采用硬掩膜方式存储在 ROM 区中，这种方式可杜绝通常 PC 中软件所遭受的病毒、木马等攻击，并且采用执行代码的访问地址受 ROM 区限制的方式，可保证代码执行的正确。

如果 COS 代码存储在 EEPROM 中(通常存放 COS 软件的补丁程序)，并且允许用户自行下载，则在下载前须严格进行代码安全性和完整性分析，避免存在软件漏洞。如不进行控制性检测，若潜在攻击者把读取所有 ROM 和 EEPROM 数据的程序做成补丁程序放置在 EEPROM 并执行，则 COS 系统完全透明。因此补丁程序需进行严格的安全性分析，并限制其读写权限。

2.5.2 协议攻击

对于运行的 COS 软件，攻击手段通常有内存异常、缓冲区溢出和逻辑错误，旨在寻找 COS 软件设计时存在的安全漏洞。此类攻击通常针对 ISO/IEC 7816-4 或者 ISO/IEC 14443-4 协议进行攻击，如：

(1) 发送错误命令格式；

(2) 大数据发送或接收块链接协议过程中，发送错误协议字节，如错误块号、错误响应字节、大数据量链接等；

(3) RFU 字节或者位的处理；

(4) 状态机转换。

COS 软件利用底层加密模块进行国际通用加解密运算和签名运算，这些安全算法的安全性通常依赖于密钥的长度，算法详细描述可参见本书第 4 章。

2.5.3 安全体系攻击

COS 软件的功能包括数据传输管理、文件管理、安全体系、命令解释。其中文件管理中，对各种类型的文件都要有一种访问机制，以存储用户数据，既要高效，又要安全。COS 的安全体系是 COS 中的重要部分。它主要是控制用户的权限，什么情况下允许外部对智能卡进行什么操作。命令解释是对行业规范中的命令和智能卡专用命令的实现。一条命令有时是多种功能的综合体，命令解释部分要对命令做出解释，根据权限满足情况，控制各种功能的实现。各种安全机制可参见本书第 5 章。

COS 软件安全体系的控制权以安全的方式移交给客户，不留任何后门，客户将利用这种安全体系保护用户数据。一旦移交安全控制权，COS 的开发商也和别人一样，对智能卡上的数据只能靠合法权限进行操作，别无他法。

系统对逻辑攻击的敏感性很大程度上取决于软件的复杂程度，安全漏洞也会随着程序代码的增加而增长。应对软件缺陷的安全策略包括：

(1) 结构化设计。以小的功能模块构建软件，使程序易于理解和校验。

(2) 正规的校验。使用数学模型来检验功能。

(3) 安全测试。对软件的运行进行安全性严格测试。

2.6 著名攻击事件

2.6.1 Mifare Classic 芯片攻击事件

Mifare Classic 芯片是目前世界上使用量最大、技术最成熟的逻辑加密卡芯片，广泛应用在学校、企事业单位、智能小区的停车场管理、身份识别、门禁控制、考勤签到、食堂就餐、娱乐消费、图书管理等场合。

2008 年 4 月，在第 24 届黑客大会上，德国学者 Henry Plotz 和弗吉尼亚大学博士 Karsten Nohl 公开了全球广泛应用的 Mifare Classic 芯片加密算法 CRYPTO1 的破译方法。同年 5 月，内嵌 Mifare 芯片的伦敦公交卡被成功克隆。该事件掀起一场大范围的安全风暴，引起全球对智能卡安全的广泛关注。

在 Mifare Classic 芯片被破解的消息传出之前，其安全性也是 Mifare Classic 芯片的忠实用户一直引以为豪的原因之一。Mifare Classic 芯片的序列号是全球唯一的，不可以更改；在读写时，卡与读写器之间采用三次双向认证机制，互相验证使用的合法性，而且在通信过程中所有的数据都加密传输；此外，卡片各个分区都有自己的读写密码和访问机制，卡内数据的安全得到了有效的保证。

关于我国应该怎样度过此次的“破解门”危机，大多数业内专家认为中国的 IC 企业应该从单纯的技术引进向发展拥有自主知识产权的技术转变，准备好从芯片、机具到系统应用的全面技术解决方案，来满足用户向 CPU 卡升级的需求，力求为最终用户提供安全、可靠、满足用户需求的 IC 卡解决方案。同时，还应加强与商业银行、银联和移动运营商等应用部门的沟通协调，共同携手提升非接触式 IC 卡应用的安全性。

2.6.2 英飞凌芯片攻击事件

前美国军事安全专家 Christopher Tarnovsky 发现英飞凌的 SLE66 CL PE 芯片存在一个安全漏洞并且在 2010 年黑帽会议上展示了他的攻击成果。英飞凌的这种芯片用于 PC、卫星电视硬件和游戏机等产品中，保护安全数据。

Tarnovsky 现在为安全公司 Flylogic 工作。他表示，破解这种具有可信赖平台模块设计的英飞凌芯片是一个很长的过程，还要使用一台电子显微镜(零售价大约 7 万美元)。这个芯片破解的计划和实施用了 6 个月的时间，包括使用酸性溶液溶解这个芯片的外壳和使用微小的探针窃听芯片的编程指令。

虽然能够物理访问这种芯片，Tarnovsky 还需要越过这个芯片的软件防御。据美联社报道，Tarnovsky 说，这个芯片是容易发怒的。如果你做错了，它就会像定时炸弹一样。

这是不是意味着英飞凌的主要安全芯片已经完全被破解了呢？英飞凌知道这种芯片的物理破解是可能的。英飞凌负责安全的一位副总裁 Joerg Borchert 对美联社说，由于这种破解需要物理访问这种芯片、一个聪明的黑客和昂贵的设备，这种风险是可以控制的。这相当于你在攻击一台计算机。

参考文献

[1] 毛丰江，温希东. 智能卡攻击技术与安全策略的研究. 计算机工程与设计，2006，27(13)：2396～2398
[2] 黄显明. 智能卡攻击技术分析及安全防范策略. 中国防伪报道，2008，(07)：42～44
[3] 马博，包斯刚. 功耗攻击对智能卡密码的威胁. 卡技术与安全，2009，(11)：46～50
[4] 郑新建，张翌维，沈绪榜. SPA 和 DPA 攻击与防御技术新进展. 小型微型计算机系统，2009，4(30)：726～731
[5] Ross J. Anderson，Markus G. Kuhn. Low Cost Attacks on Tamper Resistant Devices. In：Proceedings of the 5th International Workshop on Security Protocols. London：Springer-Verlag，1997：125～136
[6] Kai-Fan Chang. 差分错误分析技术. 卡技术与安全，2009，(11)：37～39
[7] 毛丰江. 智能卡的边频攻击分析及安全防范措施. 单片机与嵌入式系统应用，2006，(2)：9～11
[8] Mifare卡破解带来的危险以及应对方法，http://www. china. com. cn/economic/txt/2009-09/07/content_18477776. htm，2009
[9] 丁国良，郭华，陈利军等. 密码系统差分电磁分析研究. 计算机工程与设计，2009，30(12)：2892～2894
[10] Peter Laackmann，Marcus Janke. 非接触智能卡安全分析. 金卡工程，2007，(10)：74～75
[11] Hagai Bar-EI. Known Attacks Against Smartcards，http://www. discretix. com，2004
[12] 卢小东. RFID 芯片的攻击技术分析及安全设计策略. 金卡工程，2008，(1)：58～60
[13] Tarnovsky C. Deconstructing a Secure Processor. Proceedings BlackHat Conference Washington DC，http://www. blackhat. com/presentations/bh-dc-10/Tarnovsky_Chris/BlackHat-DC-2010-Tarnovsky- DASP-slides. pdf，2010
[14] Tarnovsky C. Security Failures in Secure Devices. Proceedings BlackHat Conference Europe，http://www. blackhat. com/presentations/bh-europe-08/Tarnovsky/Presentation/bh-eu-08-tarnovsky. pdf，2008
[15] Jackson W. Engineer Shows How To Crack A Secure TPM chip. Government Computer News，http://gcn. com/Articles/2010/02/02/Black-Hat-chip-crack-020210. aspx? p=1，2010



安全目标

智能卡被大量应用的一个主要原因是其具有高安全性，但随着智能卡应用范围不断扩大，针对智能卡的各种各样攻击也越来越多，且呈现日趋增长的趋势，这给整个智能卡行业造成了巨大的社会影响和经济损失。为了有效地防止攻击，智能卡系统在设计之初就应该提供若干措施来保证其安全性，也就是它的安全目标。整个智能卡系统的安全目标可分为智能卡芯片的安全目标、芯片操作系统的安全目标、应用环境的安全目标和管理的安全目标。

3.1 安全体系

针对整个智能卡系统存在的安全威胁和受到的攻击，设计者必须从整体上认识、了解和研究系统的安全策略和技术方法。

整个系统是否安全，主要取决于系统采取的安全策略和安全技术，它涉及材料学、微电子、信息论、计算机原理、密码学等多个领域，以确保信息的保密性、完整性、真实性和可用性不被破坏。整个系统安全层次分为七个层次，模型如图3-1所示。



图3-1 安全层次模型

七层之间相互联系、彼此依赖，下层向上层提供依据，上层依赖下层的支持，最终实现数据信息的安全。其中，第一、二层是法律道德规范和管理制度措施。目前世界各国政府和组织都制定了相应的严密的法律、法规和政策，以规范和制约人们的行为。同时各个使用方也都建立了满足自身需求的安全管理制度，加强管理人员的安全培训，使得整个信息安全管理更加规范化、制度化和科学化。第三、四层是物理实体和系统硬件的安全保护，主要包括自然灾害防护、设备环境安全防护和硬件本身的损伤防护。第五、六、七层是通信网络、软件、数据安全技术策略。这三层是信息系统安全的关键所在，在进行系统安全设计时应主要从数据安全保密和通信安全保密两个方面来考虑。

3.2 安全服务

开放系统互联(open system interconnection，OSI)安全体系模型中提出了普遍适用的安全体系结构，旨在开放的系统中实现进程之间安全地交换信息。

OSI 安全架构提供了安全服务和实现这些服务的安全机制。安全服务是一种由系统提供的对资源进行特殊保护的进程或通信服务。安全服务通过一种或多种安全机制来实现安全策略。

标准 X.800 定义了下列安全服务：

认证。确保通信实体就是其所声称的实体。

访问控制。保护数据免于被未经授权的实体访问。

数据保密性。保护数据免于被非授权的暴露攻击，如窃听等。

数据完整性。保护数据免于被恶意篡改、插入、删除和重发。

不可否认性。保护数据免于被信息发送方或接收方否认。

另外 X.800 还定义了可用性，并把可用性看作是与其他服务相关的性质。

上述这 6 种安全服务描述如下。

1. 认证

认证服务与保证通信的真实性有关，即接收方必须保证信息来自所声称的发送方。对于正在进行的交互，如机具与智能卡连接，就包括两方面的内容：(1)通信之初认证服务必须保证双方的真实性，一般采用外部认证、内部认证或双向认证的方式。(2)该服务必须保证通信不受哑实体的干扰，即攻击者伪装成机具或卡片进行截获数据，并发送错误数据。

认证流程详细描述参见本书第 5 章内容。

2. 访问控制

对智能卡中的各类文件和数据访问必须进行限制和控制。这些访问包括底层硬件访问、数据文件的创建/删除/锁定/解锁/读写、数据加解密处理等操作。每种企图访问的实体都必须获取权限才能进行相应的操作。

3. 数据保密性

保密性是防止通信中的数据遭到诸如窃听、流量分析等攻击，防止信息被泄漏给非授权用户。此项安全服务采用的主要方式是利用密码技术对智能卡和机具之间的信息流进行加密，使之成为密文，即使截获方获取此消息，也无法还原成信息明文本身，通过此方式来达到保密的目的。根据加密密钥不同，可分为对称加密和非对称加密。对称密钥可以使用相同的密钥进行加密，也可以使用会话密钥的方式，即每次通信都采用不同的密钥。信道加密如图 3-2 所示。

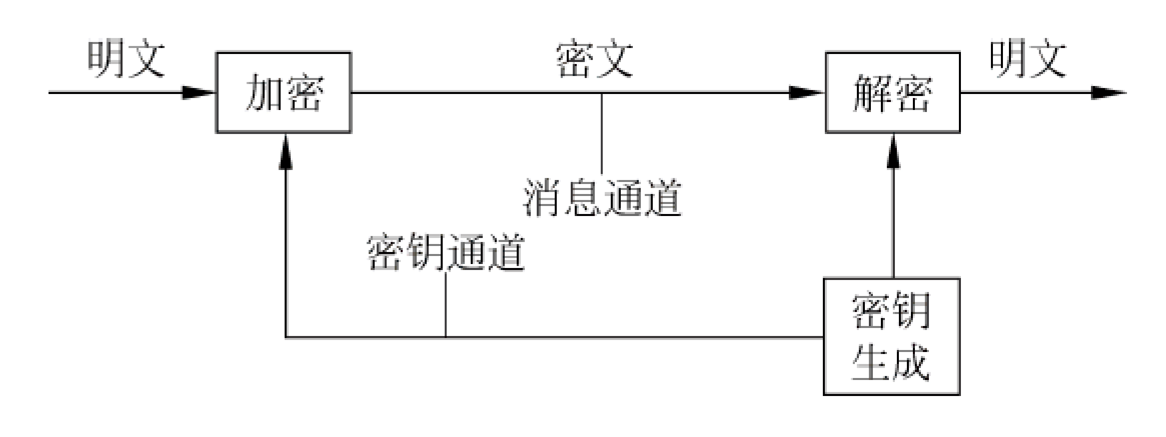


图 3-2　信道加密示意图

保密性的另一层含意是对信息存在性的隐蔽，存在性有时候比数据本身更能暴露信息，某些特定的控制访问机制有时仅仅起到隐藏数据存在性的作用。这层含意的保密性在智能卡芯片设计中体现在对某些特殊单元的保密，在操作系统中指对某些特殊指令、特殊功能单元以及特殊认证等信息的保密。

4. 数据完整性

完整性服务保证接收到的消息和发出的消息一致，不存在对消息进行复制、插入、修改、倒序、重发和破坏。为了保证所交换的信息内容不被非法修改，对信息内容进行鉴别是非常重要的。一般方法是在所交换的信息报文内加入一个报尾，称其为鉴别码。这个鉴别码是通过对报文进行某种运算而得到的，它与报文的内容密切相关，报文的正确与否可以通过这个鉴别码来检验。鉴别码由报文发送方计算产生，并和报文一起经加密后提供给接收方，接收方在收到报文后，先对之解密得到明文，然后用约定的算法计算出解密报文的鉴别码，再与收到报文中的鉴别码相比较，如果相等，则认为报文是正确的；否则就认为报文传输过程中已经被修改过，接收方就可以采取相应的措施，如拒绝接收或者报警等。在鉴别过程中，鉴别算法的设计是至关重要的。最简单的算法是计算累加和，即把所传输报文中的所有位进行累加作为该报文的鉴别码。现在流行的鉴别算法与密码学相联系，其中有采用对称算法的 MAC 机制和非对称算法的签名机制，具体的算法参见第 4 章。

消息鉴别码示意如图 3-3 所示。信息源欲发送消息 M，为了保证该消息的完整性，将 M 进行加密操作(图中的 C 操作，其中 K 为密钥)，生成安全消息摘要 $S_1=C(M)$，然后将该消息 M 和安全消息摘要 S_1 一起发送至接收方。接收方利用同样的加密操作和相同的密钥生成安全消息摘要 S_2，判断 S_2 是否和接收的安全消息摘要是否相同。如果相同，则表明消息 M 在传输过程中没有被修改。反之，该消息应抛弃。

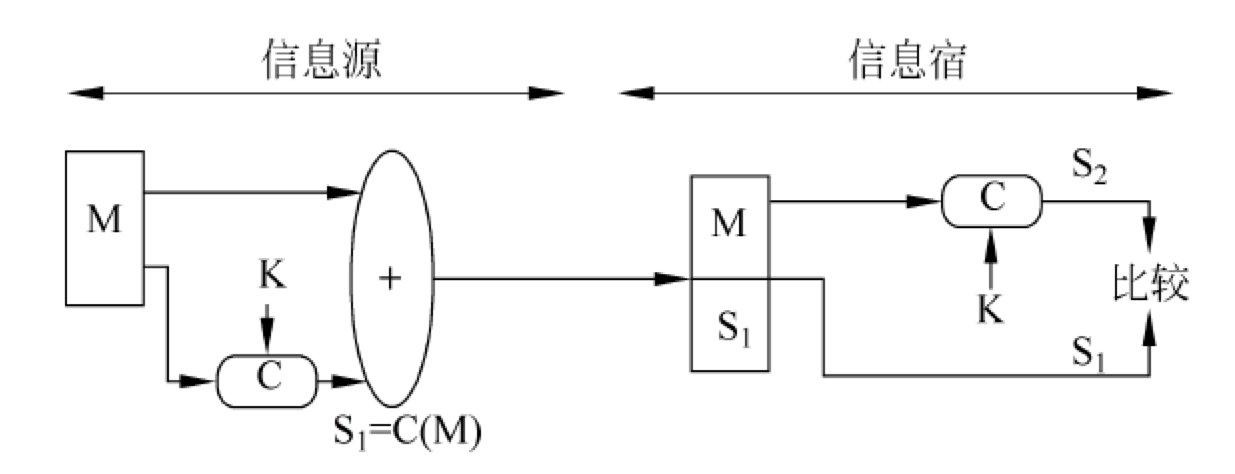


图 3-3 消息鉴别码

完整性服务可以分为有恢复功能的完整性服务和无恢复功能的完整性服务。因为对于数据完整性的破坏与主动攻击有关，所以完整性服务的重点在于检测而不是阻止攻击。如果检测到完整性遭到破坏，那么具有恢复功能的完整性服务能够恢复出被破坏的数据。

5. 不可否认性

不可否认性也称为不可抵赖性，在信息交互过程中，确信参与者的真实同一性，即所有参与者都不可否认或抵赖曾经完成的操作和承诺。利用信息源证据，可以防止发送方不真实的否认已发送信息；而利用递交接收证据，可以防止接收方事后否认已经接收到信息。一般采用 PKI 非对称密钥的方式，可以有效地防止发送方的抵赖，如图 3-4 所示。具体的工作原理参见第 4 章。

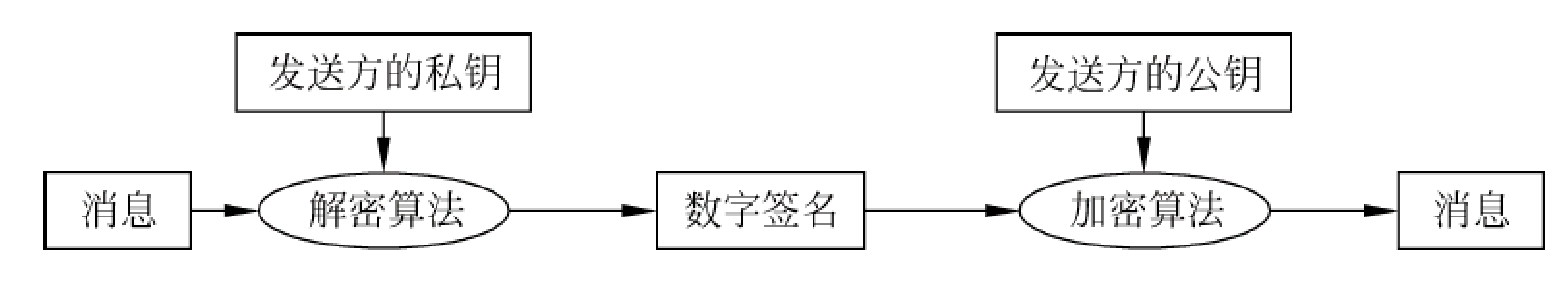


图 3-4 发送方的不可抵赖性

6. 可用性

可用性是信息可被授权系统或用户访问,并按需求使用的特性,即信息服务在需要时,允许授权用户或实体使用的特性,或者是网络部分受损或需要降级使用时,仍能为授权用户提供有效服务的特性。信息系统最基本的功能是向用户提供服务,而用户的需求是随机的、多方面的甚至是实时的。

可用性是指对信息或资源的期望使用能力。它是系统设计中的一个重要方面,因为一个不可用的系统起到的作用还不如没有此系统。而可用性之所以与安全相关是因为有人可能会蓄意地使数据或服务失效,以此来拒绝对数据或服务的访问。

从实际角度出发,世界上没有绝对安全的系统。系统安全性的选择需要在风险预期(risk appetite)和风险容忍(risk tolerance)两个因素之间找到一个最佳的平衡点。风险预期就是通过必须采取措施控制、降低的风险部分;风险容忍就是其余的风险,即可以接受的部分。

3.3 安全设计与控制

3.3.1 芯片硬件安全

在设计开发、生产制造的各个环节都需要保证芯片安全性。

1. 开发阶段

(1) 芯片的设计除了采用标准单元外,还应该按需求开发特殊的单元来实现芯片的独创性。特殊单元的设计一般是设计者独特性、创新性的体现,是与常规的设计理念不同的地方,这就增加了分析的难度和时间。

(2) 在芯片空间充足的前提下,尽量使用哑结构,该结构是半导体中不具有任何真实功能的单元,主要目的是混淆与误导攻击者,其安全性在于对哑结构存在以及相应位置的保密。哑单元可以受到监测,并在受到攻击的情况下启动相应的预防措施,例如切断电源或擦除数据等。

(3) 芯片总线用来连接处理器与各个存储器,是芯片内的通道。总线的设计应该采用静态或特有的方式扰乱排列,使得攻击者无法从外部获知各条总线的功能。图 3-5 表示了芯片的普通数据总线,图 3-6 为扰乱后的数据总线。

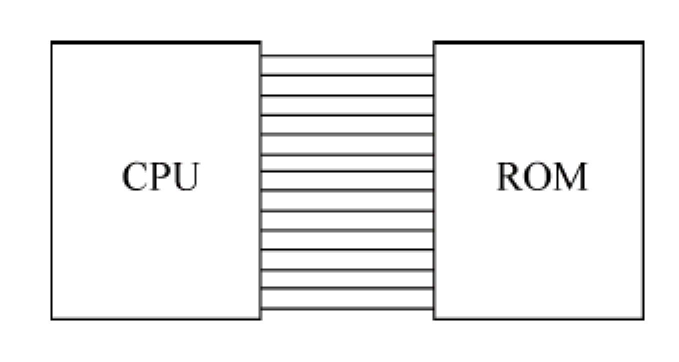


图 3-5 普通数据总线

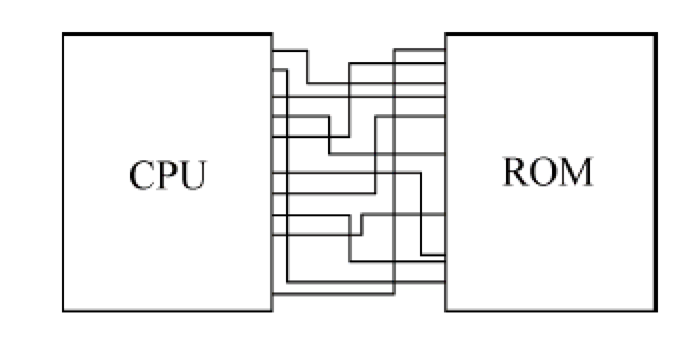


图 3-6 扰乱的数据总线

(4) 存储器的设计应该能够防止可见光、紫外线,防止芯片在使用选择性蚀刻半导体方法时内容的暴露。现在高级的芯片设计多采用存储器分层设计的概念,例如:

- EEPROM 分散到各个电路层上。
- 将各个不同的功能单元混淆在一起。

- 存储器地址采用混乱的方案，如果没有相关的信息，攻击者很难分析出存储器是如何编址的。存储器地址的线性增加和混乱如图 3-7 所示。

00	01	02	03	04	05	06	07	
10	11	12	13	14	15	16	17	
20	20	22						
30	31							
40								

⇨

31	12	02	21	15	05	11	40	
10	06	01	13	14	04	16	20	
17	03	22						
30	00							
07								

图 3-7　存储器地址线性增加和混乱

- 可以将 CPU、协处理器、逻辑电路进行混淆，如图 3-8 所示。

(5) 芯片的表面应该覆盖一层金属化的导流层，以防止通过测量芯片一定区域内的电压来推导出 RAM 的内容。如使用化学腐蚀的办法去除掉该层，则芯片停止工作。图 3-9 显示了芯片表面的导流层，图 3-10 显示了芯片遭到攻击后信号发生变化的情况。

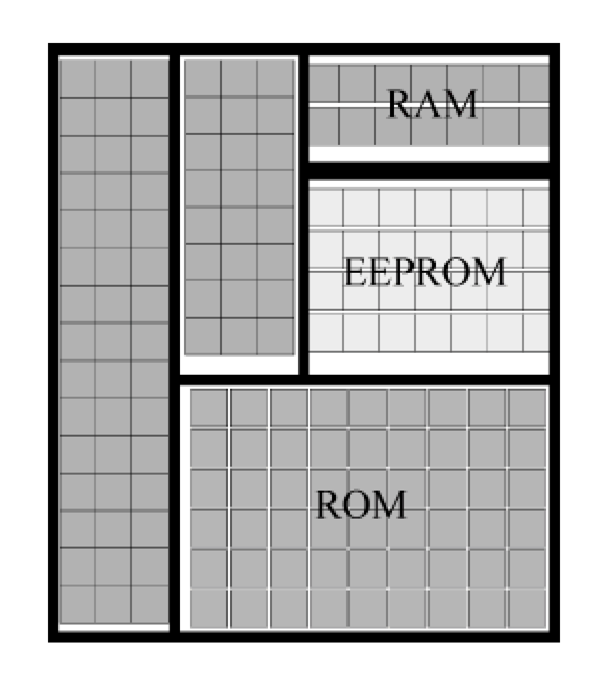


图 3-8　将 CPU、协处理器、逻辑电路混淆的芯片结构

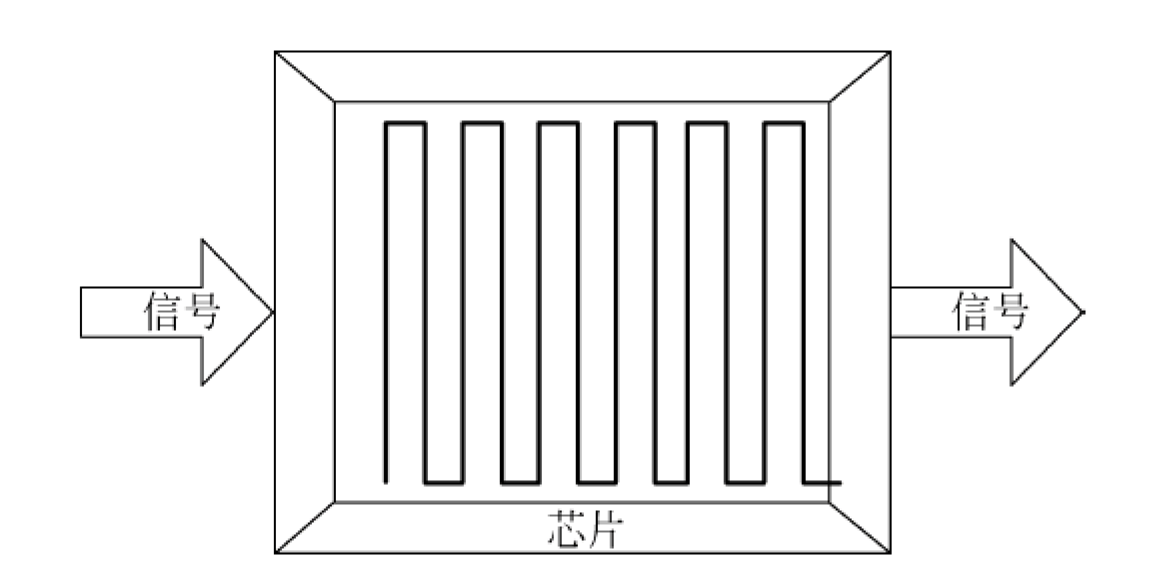


图 3-9　芯片表面的导流层

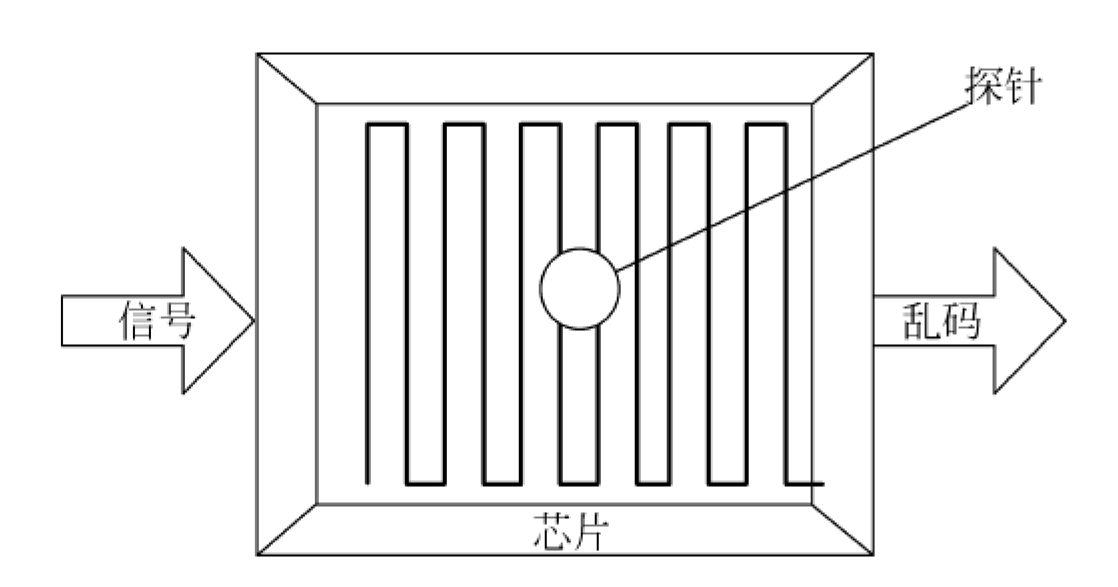


图 3-10　芯片遭到攻击后的现象

(6) 芯片的存储器设计应该具有存储器数据加、解密的功能，这种防御的基本思想是攻击者即使获得存储器的内容，也无法对内容解析。图 3-11 显示了芯片架构加密前后的变化，图 3-12 显示了 EEPROM 数据进行加密的变化。

(7) 处理器的设计应该保证对所有机器命令具有基本相同的电流损耗，这样可以防止简单功率分析(SPA)和差分功率分析(DPA)，有关 SPA 和 DPA 的具体内容请参见第 2 章内容。

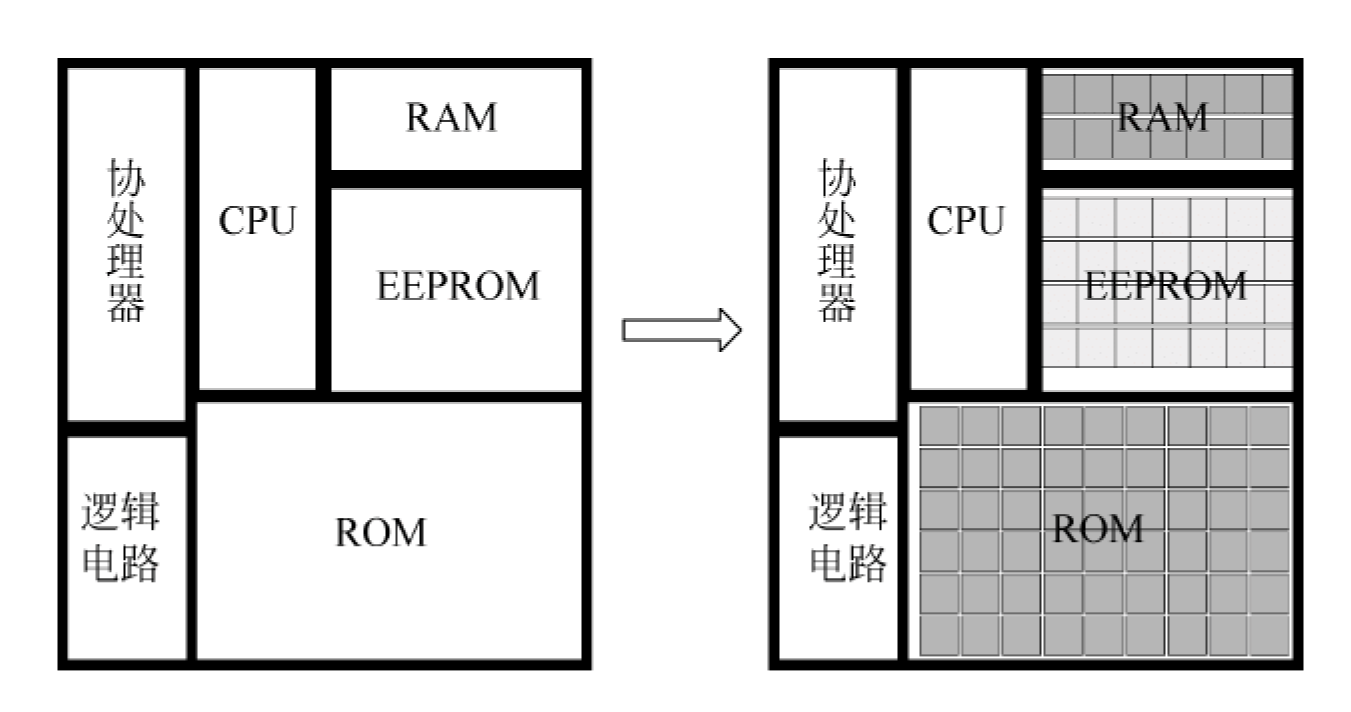


图 3-11 普通的芯片架构和加密后的芯片架构

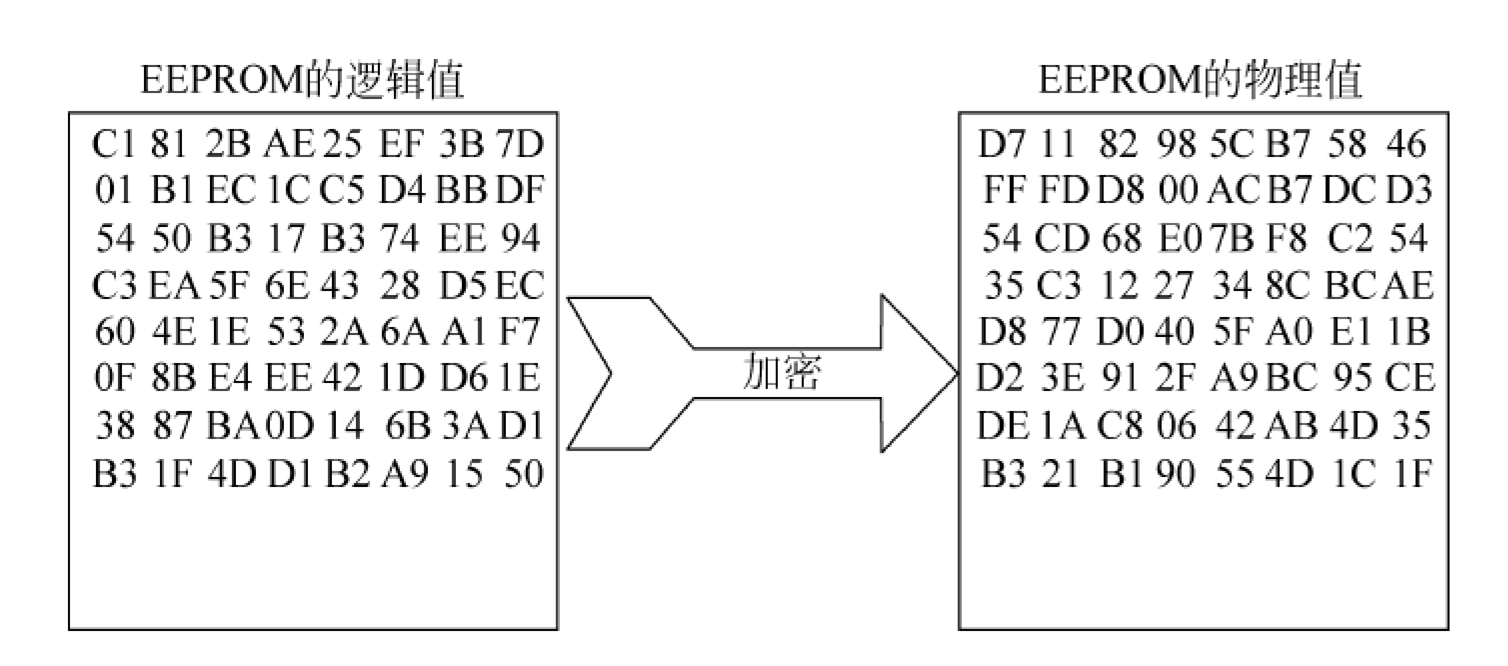


图 3-12 EEPROM 加密示例

2. 制造阶段

在制造智能卡芯片时，应确保物理封装的坚固性，并且必须做到能够承受相应的作用力、静电放电、静态电场、静态磁场、交变电磁场、红外线照射、化学物质腐蚀、弯折、钢笔刻画、盖章等，这方面的安全目标与智能卡的具体设计方案和所采用的材料、设备、生产线上的操作员以及工艺有关。制造阶段的安全目标如下：

（1）芯片的制造设备应为特定的并且复杂昂贵的设备，操作工人都需要具备各种专业的知识和技能，这样可使芯片伪造工作必须投入大量的资金才可以实现，如此增加伪造的成本，使伪造变得无利可图。芯片制造如采用光学平晶、集成电路硅基片的加工，采用精密研磨和抛光，可达到纳米级的水平，采用 LIGA 技术可以达到 0.1μm 的线宽，采用电子束、离子束、X 光光刻，则可达到数十纳米到纳米级的线宽。这些高精尖的技术和相应的设备只有少数几个专业的公司掌握。

（2）芯片的制造过程是一个复杂的系统工程，需要经过上百套工序才可以完成，每个工序的具体参数都应该严格保密。整个过程主要包括：

① 芯片设计；

② COS 的设计与开发；

③ 掩膜和半导体芯片的制造；

④ 晶圆上的芯片测试；

⑤ 剪薄划片；

⑥ 芯片安装到条带上；

⑦ 芯片的压焊；

⑧ 模块的封装；

⑨ 模块的测试。

(3) 芯片制造尽量采用小的引线宽度工艺。IBM 及其合作伙伴在 2008 年 8 月表示，IBM 将在 2011 年量产 22nm 工艺的芯片。目前智能卡芯片所用工艺逐渐从 250nm、180nm 逐步提高到 90nm、65nm 的新工艺。图 3-13 展示了 Wafer 盘片的外形模样。

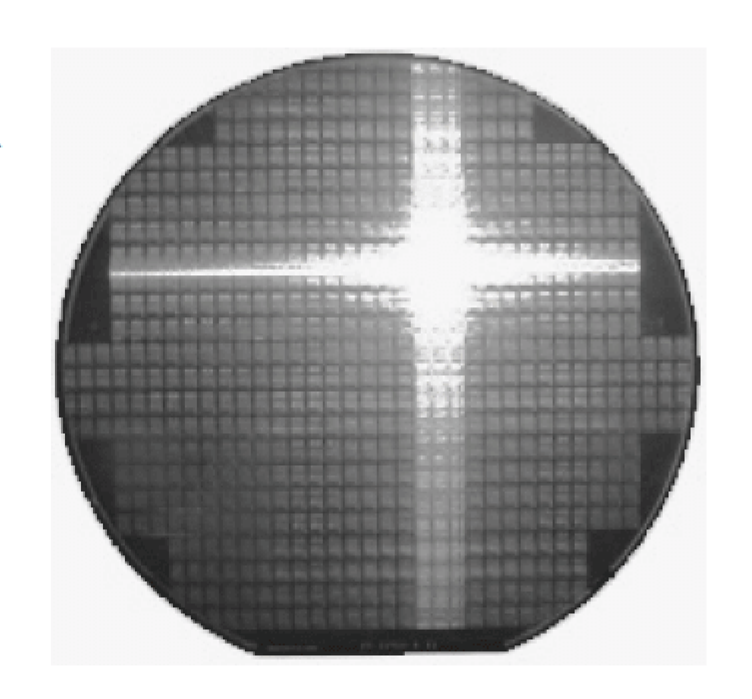


图 3-13 Wafer 盘图片

(4) 芯片制造过程结束时，在硅片的表面应该加盖一层钝化层，这可以阻止芯片表层的氧化和其他化学反应。芯片的钝化层和芯片与钝化层间介质的不透明性，使得攻击者难以观察到下层金属化机构，难以接触机械探针与金属化层，也无法测得芯片内部节点的电压和波形；钝化层的不导电性、介质层的荷电效应，使得攻击者即使采用扫描电镜和电子束测试技术进行分析，也难以获得正确的图像。

(5) 芯片应该具有电源监控器来监测芯片的工作电压，如电压超出芯片稳定工作范围就按规定关闭芯片，使芯片不能在边界情况下工作(芯片此时已不能正常工作)。如果没有电源监控器，芯片工作在供电范围之外，就会出现程序计数器不稳定的情况，从而导致程序错误跳转，出现运行错误，造成未知的结果。

(6) 芯片应该具有频率监控器，该监控器在芯片的工作频率为某区域值(举例如 1～5MHz)，当工作频率超出该区域值时，芯片停止工作，这就防止了外界控制芯片在单步执行模式下运行，使对工作电流消耗的测量和芯片表面的电位测量变得更为困难。图 3-14 表示了频率监控器的应用模式。

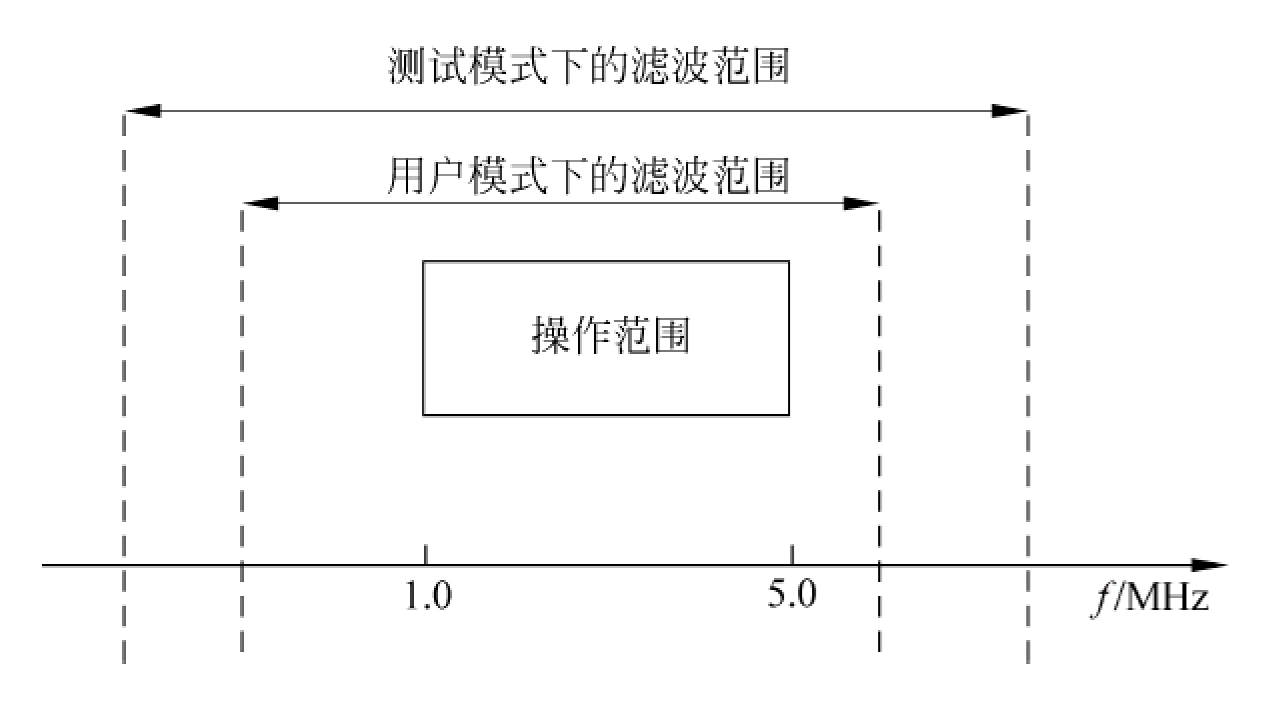


图 3-14 频率监控器

(7) 芯片制造结束时应该使芯片从测试模式转换成用户模式，并且该过程必须为不可逆操作。由于在测试模式下，不可避免地要从 EEPROM 中读出数据，所以在用户模式下，要绝对禁止此类操作的可能性。转化操作一般采用多晶体硅熔断丝来实现。现在比较流行的方式为在芯片制造时将 CPU 的引线全部引出到芯片附近以便测试，这些引线在晶圆划片时(转化到用户模式)都被锯掉，以目前的技术还不能在被锯断的点上连接，这就杜绝了测试的可能性。

(8) 一旦产品开发完成，整个源代码、设计图纸一般都需要提交到第三方独立的测试结构进行检测。这项测试花费大量金钱和时间，执行它的主要目的是为了检查产品设计中存

在的漏洞，以及设计者可能在产品中故意隐藏的木马，从而保证产品的安全性。

3.3.2 芯片软件安全

1. 复位后对软件和硬件进行测试

在智能卡上电、复位操作之后，其芯片操作系统必须规定程序进入指定初始状态，此目的是防止攻击者操纵智能卡，使其处于未定义的状态；此外还应该对硬件的重要部分进行测试，以了解它们是否处于正常工作状态，例如对 RAM 的检测，程序运行时的大多数变量都存在此处，其中任何一位遭到攻击都会使智能卡处于未知的状态。

2. 防缺陷插入

芯片操作系统必须能抵御插入缺陷数据重复探测的攻击，对错误的命令和未知的数据需要返回对应的错误代码，并进行合理的响应和处理。由于操作系统属于被动操作，不能主动选择处理的数据，只能被动进行响应，所以该系统设计必须要有很强的鲁棒性，能够处理所有的数据。

3. 防重放攻击

芯片操作系统应提供诸如鉴别机制等功能以抵御重放攻击，当鉴别操作已成功结束或被中止时，攻击者无法通过重放或重新启动此项操作来达到非法获取智能卡内数据的目的。重放攻击主要指攻击者利用其消息的再生能力，生成诚实用户所期望的消息格式并重放，从而达到破坏的目的。防止消息重放攻击的手段是保证信息的新鲜性，保证消息是刚刚被产生和发送，而不是先前产生和发送的消息的重复。这部分的防护一般可以采用时间戳和挑战应答机制实现。

4. 访问控制

为了保证数据的保密性和信息不被随意获取，以及防止信息的泄漏，芯片操作系统必须针对存储在 EEPROM 中的文件和数据设计控制规则。

在整个智能卡不同的生命周期内使用者、管理者、发行者等都要对自己掌握的数据进行控制，对不同的文件和操作采用不同的控制规则，例如文件的建立、锁定、解锁、删除等。

访问控制一般采用认证的方式实现，认证的基本思想是通过验证称谓者（人或事）的一个或多个特有参数的真实性和有效性，来达到验证称谓者是否名副其实的目的。一般采用以下几种认证方式，具体描述可参考本书第 5 章。

（1）PIN 认证。

（2）生物特征认证。

（3）基于挑战-响应认证模式。

（4）基于 PKI 的认证模式。

5. 生命周期管理

整个生命周期为智能卡的应用生命周期，该生命周期包括五个阶段，分别为创建期、安装期、使用期（有效）、使用期（无效）和终止期。

芯片操作系统必须提供控制或限制在特定生命周期内使用特定命令的方法，在其他阶段内这些命令应该被禁止。例如，创建期的命令在使用期内无法使用。

芯片操作系统必须对生命周期各阶段进行管理控制，生命周期的每一阶段都用文件状态被唯一标识，并可通过标识信息追溯到生命周期各阶段。

6. 密钥管理

智能卡能被大量应用的一个重要的原因就是其安全性，而安全性在很大程度上取决于密钥，所以芯片操作系统必须具有安全管理密码的功能，必须符合国家及行业或组织的密钥管理相关标准或规范。密钥文件绝对不能从芯片读出，只能对明文数据进行运算，数据加密后输出密文；对接收的密文进行解密，输出明文。

7. 配置控制

配置管理是通过技术或行政手段对操作系统及其开发过程和生命周期进行控制、规范的一系列措施。配置管理的目标是记录操作系统的演化过程，确保开发者在生命周期中各个阶段都能得到精确的产品配置。

配置管理对于提高团队开发效率、保障软件产品质量具有重要意义。配置管理贯穿于整个操作系统开发活动的始终，覆盖了开发活动的各个环节，全面地管理各个配置项，监控各配置项的状态，并向相关的人员报告，从而实现对软件过程的控制。

8. 应用级防护

(1) 采用随机处理顺序来减少相关的信号。例如，算法中的平行置换可依随机的顺序来完成，置换的数目重新排序，则可将一次置换产生的信号分解。

(2) 利用随机延时和改变路径来增加计时噪声。计时噪声会妨碍轨迹的排列，并降低差分轨迹的质量。

(3) 消除密钥值及中间媒介值的时间依存性。

(4) 用随机值来隐蔽中间媒介值。能量的泄露取决于一个数据中的位数。如果在实际数据上加上随机数据，处理完之后再减去，那么传递的路径将不会泄露有用的信息。不过，这种隐蔽将会导致传递函数的非线性并产生错误的结果。因此，这些函数需要仔细地重新设计以补偿由随机数据引起的背离。

(5) 通过检查关键的程序流向以及加密运算结果来实现故障监测。求两次运算结果并加以比较是检测结果有效性的方法之一，但若两次都注入同样的错误，则无法检测出来。因此最佳的方法是由结果反向运算求出其输入，并与原来的输入进行比较。

(6) 重设计数器，用于限制攻击者试探的次数。连续 3 次 PIN 校验失败后自锁是防范差分功耗分析的有效方法。

(7) 限制加密算法中输入输出的控制和可见度。如果只能选择部分输入，或只有部分算法的结果返回，攻击者就无法完成差分功耗分析。

3.3.3 应用环境安全

1. 设计控制

芯片和芯片操作系统的设计版图、安全机制细节、概要设计、详细设计、使用说明、实现表示等信息只公开给相应的人员，严防泄露给其他无关人员。开发过程中形成的任何过程文档、技术方案、代码等都需要进行严格的控制，最大程度保证安全。开发环境应该能保证产品在开发过程中不被非授权查阅和篡改；开发过程中使用的大量的开发工具、编译软件，都可能向攻击者提供相关的信息，使其能够获知安全系统的工作原理，所以也要确保开发工具在产品的开发过程中不被非授权使用，如图 3-15 所示。

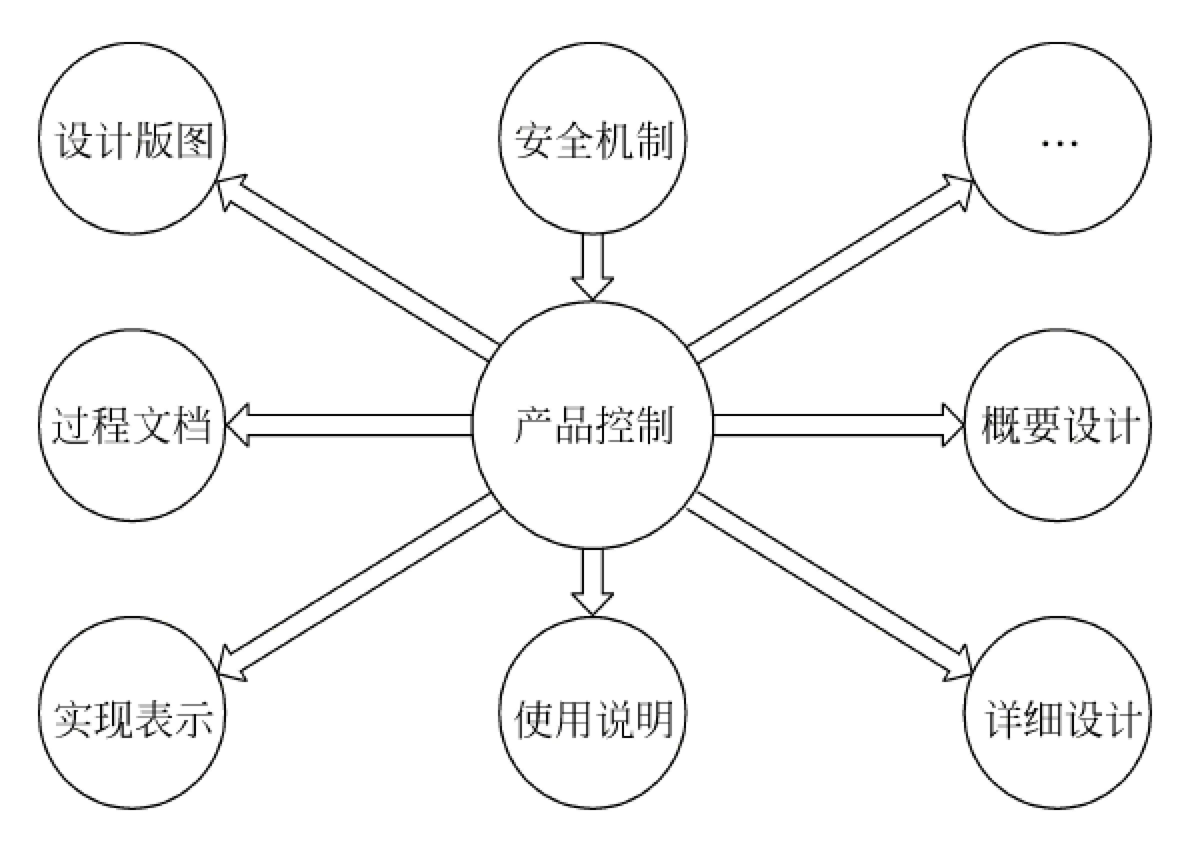


图 3-15 设计控制示意图

2. 交付过程

确保芯片和芯片操作系统在开发完成后递交给生产厂家过程的安全和保密，以及产品在交付给初始化中心、个人化中心以及最终交付给用户的整个过程的安全和保密，并对整个过程做详细的记录，所有参与人员都要进行必要的技术培训，使其具备必要的知识和技能，以满足交付过程的要求。产品的交付过程参见图 3-16。

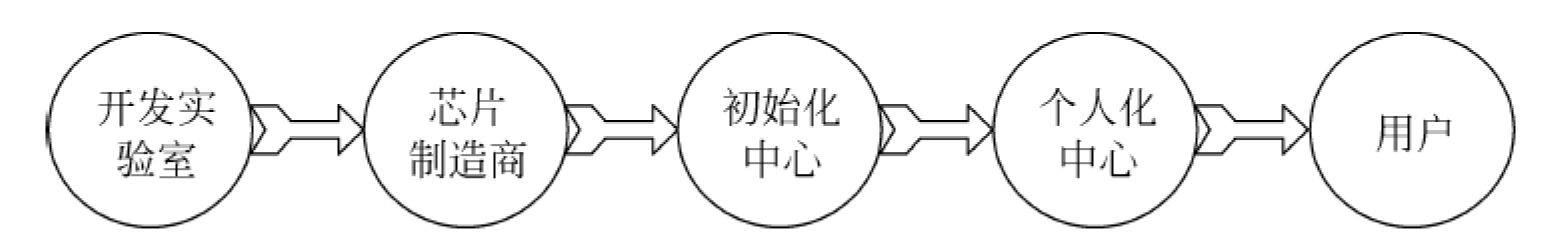


图 3-16 产品的交付过程

3. 初始化数据的保护

初始化系统将专用密码设备生成的密钥，经处理后注入到每一张智能卡芯片中，替换其在芯片制造商那里生成的传输密钥，同时为每张空白卡建立文件系统和访问密钥文件系统。系统还负责记录卡的初始化生产过程，统计生产数据。经初始化的卡，利用内含密钥的控制实现身份认证和对芯片内不同数据区的存取权限控制，以实现对智能卡数字信息的安全访问功能。

只有初始化中心的工作人员在授权的情况下才能访问初始化数据，并且保证初始数据不外漏出初始化中心。

4. 密钥控制

密钥管理的安全需求涵盖了密钥的整个生命周期，包括密钥产生、密钥注入、密钥存储、密钥使用、密钥导出的安全，确保密钥在任何情况下均不以明文形式存放于芯片之外，即使在芯片内部，密钥的明文也要存放在 RAM 中，掉电后自动清除，以防止对硬件进行扫描攻击，最大程度地防止来自外部或内部对密钥泄漏的威胁。

密钥管理系统采用层次结构，数据传送实际使用的是由主密钥按某种协议产生的过程密钥，而且仅对一次交易有效。卡中的密钥文件是严格保密的，用户既不能读出也不能写入，更不能删除。读写器一般使用安全存取模块(secure access module，SAM)来存放密钥并实现加解密算法。

3.3.4 管理安全

现在越来越多的企业依赖于信息技术，创建敏捷的信息基础设施——信息系统，以提高

响应市场需求的能力、鼓励企业创新并保证企业的竞争优势，来追求最大商业利润、降低生产成本、提高服务的质量、建立与商业伙伴及客户之间的密切关系。当企业要充分依赖信息系统实现以上目标时，管理者就必须要对企业实施安全管理，目的在于实现以下目标：

（1）保证数据安全，保证企业有价值和敏感的信息资源不被泄漏或窃取。

（2）提高效率和性能，保证企业商业信息系统的稳定可靠运行，进而提高管理的效率。

（3）赢得信心，提高合作伙伴、客户等与企业从事商务往来的意愿和信心。

（4）博得名声，提高客户和合作伙伴的客户满意度，树立企业良好形象。

1. 开发环境必须安全

在芯片的开发阶段，一般来说能够接触到核心代码、设计版图等的人员有限，对此方面的攻击需要特别专业的设备以及高专业水平的专家才可以完成，所以此阶段的攻击相对比较少。不过成功攻击的潜在危险性在此阶段仍然存在。

芯片的开发一般都是在特殊保密实验室完成，与外界进行物理隔离，并且采用完备的安全监控措施进行安全保护，例如采用监听器、泄露电缆报警器等措施，典型的安保措施参见图 3-17。

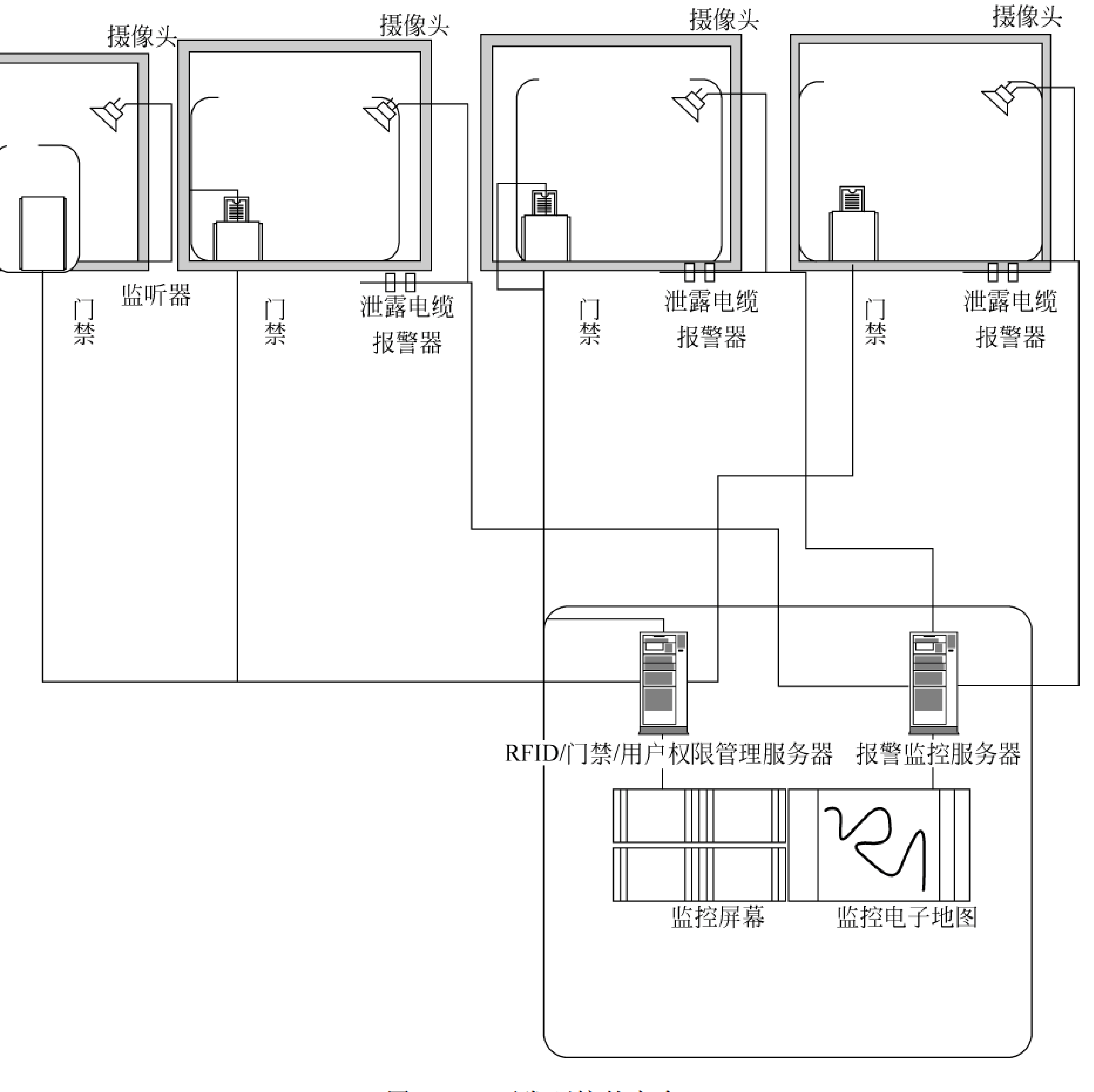


图 3-17　开发环境的安全

通常采用的开发安保措施具体有：

(1) 设计人员使用的计算机通常是一个独立的网络，与外部隔离。

(2) 严格禁止使用软盘、移动存储设备。

(3) 禁止使用具有摄像头功能的手机。

(4) 严格禁止将设计的图纸、材料带出实验室。

2. 设计过程管理

设计中需要坚持一系列的基本准则，例如：

(1) 防止攻击的各种传感器必须要能够真正工作。

(2) 绝对不允许无说明的机制或功能出现在芯片上，即使这些芯片的特性很难被检测出，但由于以后可能被用于攻击，尽管对开发者来说非常有帮助，也必须严格禁止。

(3) 设计人员绝不能单独承担一个项目，至少有两个开发者在一起工作，这样完成的设计融合了多位开发者的智慧和经验，也经得起推敲与攻击，有效地防止了内部人员的攻击。此外，需要对开发代码进行严格的审查，这样既保证了代码的质量，同时也对开发过程进行了严格的控制。

3. 人员管理

加强对人员的安全教育和管理，确保内部信息不外漏，并且保证没有人能够完整地掌握整个生产的所有环节，在一定程度上防止内部犯罪的发生。定期对员工进行安全和保密教育，使保密的理念深入到每个员工的心里，时刻按照此制度进行操作。

4. 环境管理

加强对制造环境的监测和监控，保证各个环境参数都满足芯片制造的要求，芯片生产对整个环境有极其苛刻的要求，环境是否符合要求会影响到生产中的每个环节，如果出现不符合则会严重影响产品的质量；对操作人员的操作行为进行记录，对整个生产流程进行有效的控制和管理，控制每道工序的质量，努力提高企业的现代化和规范化管理。图 3-18 是对生产环境进行监控的展示。



图 3-18 制造环境的监测

3.4 安全架构

智能卡应用系统的安全性由所有有关的诸多关键因素、环节所决定，所以必须全面而综合地考虑，不可缺漏，安全架构如图 3-19 所示。



图 3-19 系统的安全架构

从传统的安全意义上讲，要求做到通信的安全性、网络的安全性；从业务的角度而言，要求做到操作的安全性、数据的安全性；从运行的角度而言，要保证系统每天 24 小时不间断运行，要求做到系统的高稳定性和高可靠性；就 IC 卡系统本身的特殊性而言，要求系统做到密钥的安全性、卡片的安全性、终端的安全性；就信息系统本身的特点和日常管理而言，必须加强病毒的防范，随时更新病毒库，保证将病毒拒之门外。只要是可能影响系统安全、稳定、持续运行的一切不利因素都必须要严格地防范和控制。通过信任、认证和访问控制、防病毒等安全措施，在安全基础平台上为整个系统提供安全核心，也为安全设计和安全操作提供底层保证。

参考文献

[1] 傅锦伟，许海成. 信息安全的策略和安全防范技术. 蒙自师范高等专科学校学报，2001，3(6)：34～35
[2] 刘建伟，王育民. 网络安全——技术与实践. 北京：清华大学出版社，2005
[3] 傅光轩. 漏洞扫描技术研究. 贵阳：贵州大学，2006
[4] 王卓仁，王锋. 智能卡大全——智能卡的结构、功能、应用. 第 3 版. 北京：电子工业出版社，2002
[5] 朱剑英. 关于制造科技与制造业发展战略问题的思考. 机械制造与自动化，2007，36(1)：1～9
[6] 王向东，陈咏梅，王守觉等. 基于神经网络的集成电路生产过程建模与优化. 自动化学报，2007，27(3)：289～294
[7] 翁寿松. 65nm 芯片设计和制造中的几个问题. 微纳电子技术，2006，(7)：319～322
[8] GB/T 20276-2006，信息安全技术. 智能卡嵌入式软件. 安全技术要求(EAL4 增强级). 北京：中华人民共和国国家质量监督检验检疫总局 中国国家标准化管理委员会，2006
[9] 胡声州，罗南，卢震辉. 安全协议重放攻击的关联性分析. 网络通讯与安全，2001，3(6)：332～354
[10] 张毅，任宇. 基于高速安全芯片的安全策略设计. 信息安全，2007，23(4-3)：73～75
[11] 张利华，朱灿焰，张其善. 智能卡及其应用技术研究. 微型机与应用，2002，(12)：4～6

[12] Biham E and Shamir A. Differential Fault Analysis of Secret Key Cryptosystems. Crypto' 97, Springer-Verlag, 1997
[13] Schindler W. A Combined Timing and Power Attack. Public Key Cryptography, 2002, 2274: 263～279
[14] Ludger Hemme. A Differential Fault Attack against Early Rounds of (Triple-) DES. Cryptographic hardware and embedded system, 2004, 3156: 254～267
[15] 冯清枝，王志群. 智能卡的安全机制及其防范策略. 中国人民公安大学学报，2004.(1): 95～97
[16] 岳配，孙冬梅，张大伟. 智能卡数据交互安全性的研究. 计算机安全，2009，(6): 60～62



安全算法

智能卡主要应用在安全应用中，从芯片设计、COS设计到上层应用设计，安全算法贯穿始终。在芯片中，安全算法可用两种方式实现：硬件实现（通过控制协处理器来完成）和软件实现。PC下的应用层则主要利用软件完成。无论利用哪种方式实现，深入理解安全算法原理与实现都是非常重要的。

在智能卡应用中，常用的安全算法有：

（1）属于对称密钥体制的加解密算法：DES/3DES算法、AES算法。

（2）属于非对称密钥体制的签名算法：RSA算法、ECC算法。

（3）计算摘要或消息认证码的算法：SHAx系列算法、MAC算法。

4.1 DES/3DES算法

目前在智能卡中应用较多的加密技术基本上是对称密码体制，其中较典型的加密算法是DES（data encryption standard）算法或延伸算法如三重DES（triple DES，3DES）。

1972年，美国商业部所属国家标准局（national bureau of standards，NBS）开始了一项计算机数据保护标准的发展规划。NBS在1973年5月13日的联邦记录FR1973中公布了一项公告，征求在传输和存储数据中保护计算机数据的密码算法的建议。DES密码系统是由IBM公司的W. Tuchman和C. Meyer于1971—1972年研制成功的。该算法于1975年3月公开发表，于1977年被美国国家标准局批准作为美国数据加密标准。DES算法完全符合美国国家标准局提出的有关数据加密的4个要求：

（1）提供高质量的数据保护，防止数据未经授权的泄漏和未被察觉的修改。

（2）具有相当高的复杂性，使得破译的开销超过收益，同时又要便于理解和掌握。

（3）DES密码体制的安全性应该不依赖于算法的保密，其安全性仅以加密密钥的保密为基础。

（4）实现经济，运行有效，并且适用于多种完全不同的应用。

4.1.1 算法描述

DES算法属于分组密码算法。分组密码算法是将明文消息编码表示后的数字序列 x_1，$x_2 \cdots (x_i = 0$ 或 $1)$，划分成长为 m 的组 $X_1 = (x_1, x_2, \cdots, x_m)$，$X_2 = (x_{m+1}, x_{m+2}, \cdots, x_{2m}) \cdots$，长为 m 的各组向量分别在密钥 $K = (k_1, k_2, \cdots, k_n)$ 的控制下变换成等长的输出数字序列 $Y = (Y_1, Y_2, \cdots, Y_n)$（长为 n 的向量）。

DES使用长度为56位的密钥以及附加的8位奇偶校验位，加密64位的明文，产生长度

为 64 位的密文。DES 加密算法的步骤如下：

(1) 将明文分组，每个分组输入为 64 位的明文。

(2) 初始置换(IP)，初始置换过程是与密钥无关的操作，仅仅对 64 位码进行移位操作。

(3) 迭代过程，共 16 轮运算，这是一个与密钥有关的对分组进行加密的运算，DES 轮计算如图 4-1 所示。

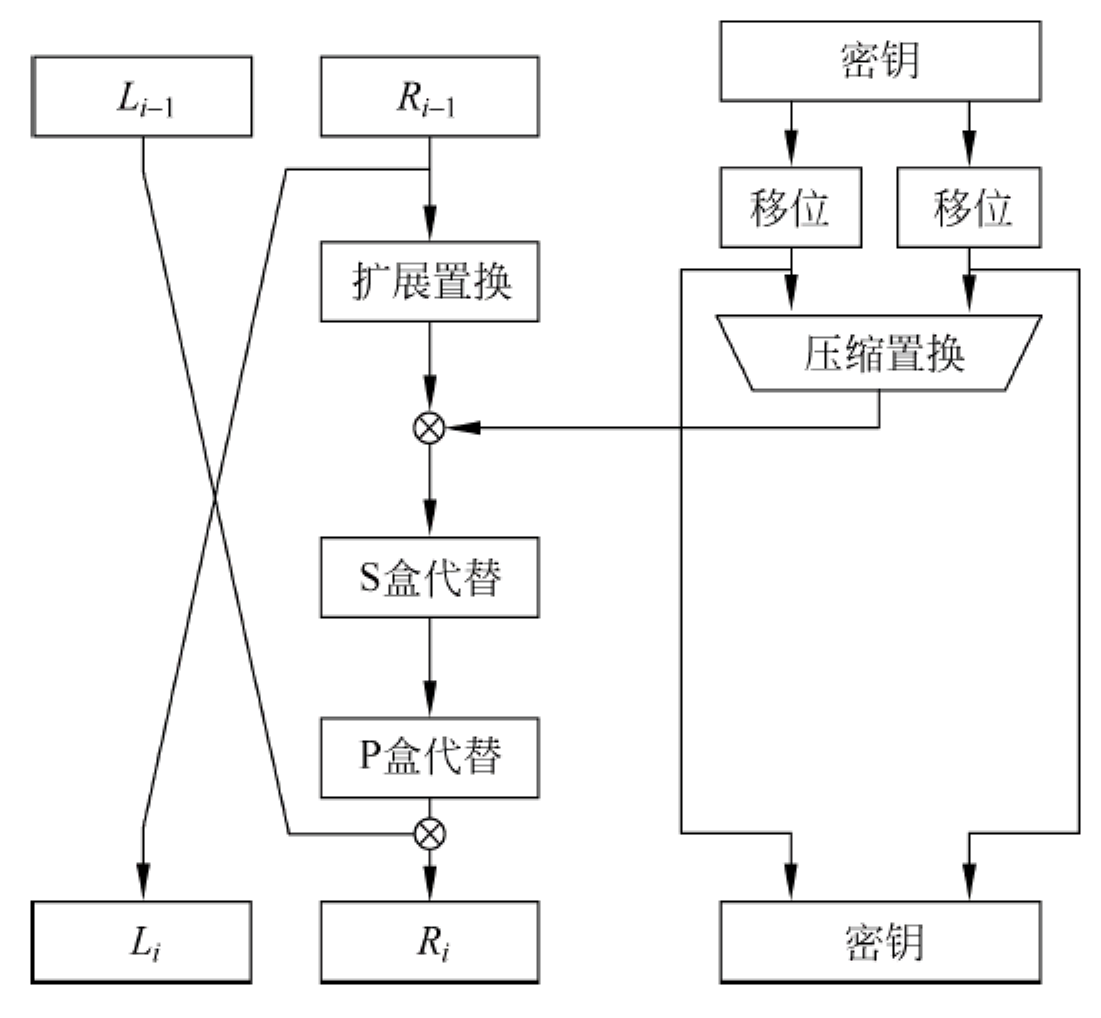


图 4-1 一轮 DES 运算过程

(4) 逆初始置换(IP^{-1})，这一变换过程也不需要密钥。

(5) 输出 64 位的密文。

DES 使用 16 个循环。DES 算法处理的数据对象是一组 64 位的明文串。设该明文串为 $m=m_1m_2\cdots m_{64}$($m_i=0$ 或 1)。明文串经过 64 位的密钥 K 来加密，最后生成长度为 64 位的密文 E。

DES 算法应用已经非常广泛，网上资源非常丰富，本书不再赘述其实现细节，如有需要请参考相关标准或网上资源。

攻击 DES 的主要形式为暴力密钥搜索，即重复尝试各种密钥直到找到正确密钥为止。如果 DES 使用 56 位的密钥，则可能的密钥数量是 2^{56} 个。随着计算机系统能力的不断发展，DES 的安全性受到严重挑战，单 DES 算法在智能卡应用中将逐渐不再使用。

3DES 主要是为了增加 DES 的有效密钥长度，使之更安全。3DES 使用 168 位密钥对数据进行三次加解密运算，提供更强大的安全性。3DES 共有三个密钥 K_1、K_2 和 K_3，如果 $K_1=K_3$，则称为双密钥 3DES，如果三者都不同，则称为三密钥 3DES。

3DES 的加密过程：用 K_1 加密，用 K_2 解密再用 K_3 加密；其解密过程：用 K_3 解密，用 K_2 加密，再用 K_1 解密。加解密过程见图 4-2。

4.1.2 分组模式

DES/3DES 算法属于分组算法，分组算法通常有多种常用的块加密模式，其中在智能卡中最常用的是电子密码本模式(electronic code book，ECB)和密码块链接模式(cipher block chaining，CBC)。

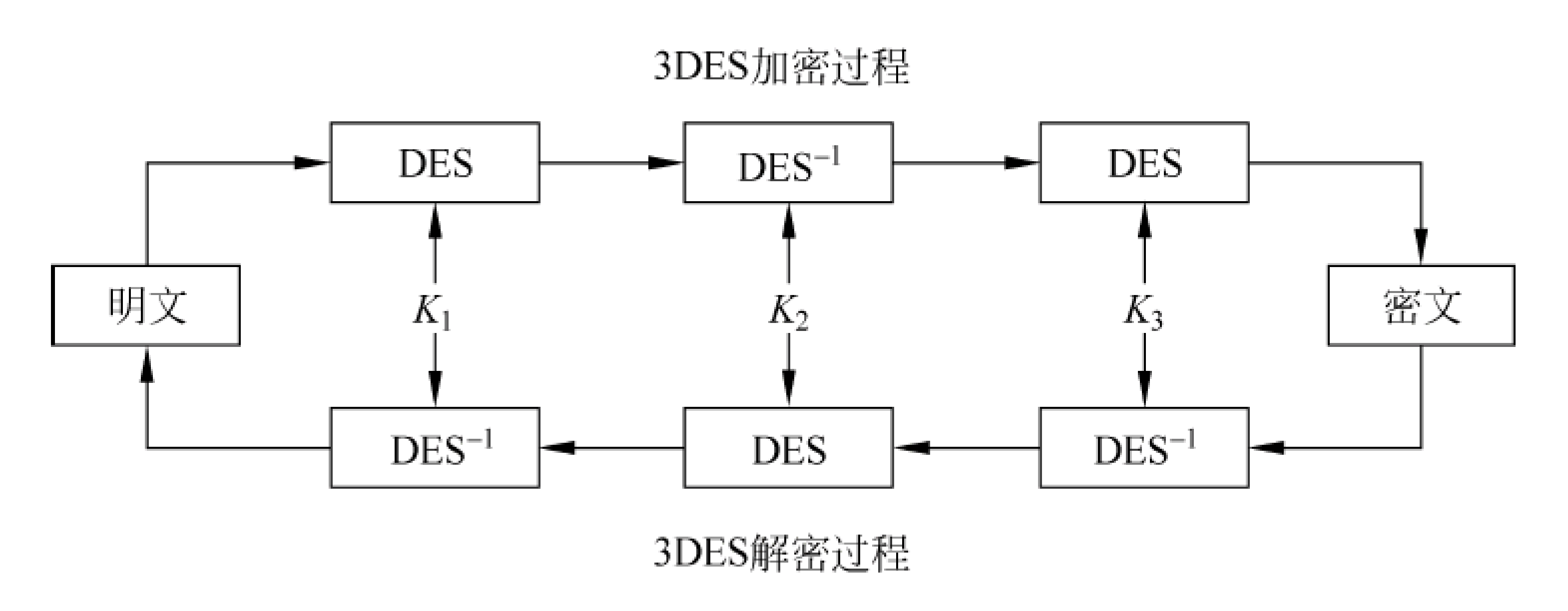


图 4-2　3DES 加解密过程

1. ECB 模式

在 ECB 模式中，每块明文都是独立于其他块加密的。ECB 模式可以高效地并行多个数据块的加密。因为 ECB 模式对相同明文块的加密总是产生相同的密文块，易受到密码分析的攻击，所以不适合保护敏感数据。

ECB 模式的加密过程描述如下，参见图 4-3。

X_i 为明文块 i，Y_i 为密文块 i，K_s 为密钥

$$Y_i = E(K_s)[X_i], \quad i = 1, 2, \cdots, n$$

$$Y = (Y_1 \mid\mid Y_2 \mid\mid \cdots \mid\mid Y_n)$$

ECB 模式的解密过程为

$$Y_i = D(K_s)[X_i], \quad i = 1, 2, \cdots, k$$

$$Y = (Y_1 \mid\mid Y_2 \mid\mid \cdots \mid\mid Y_k)$$

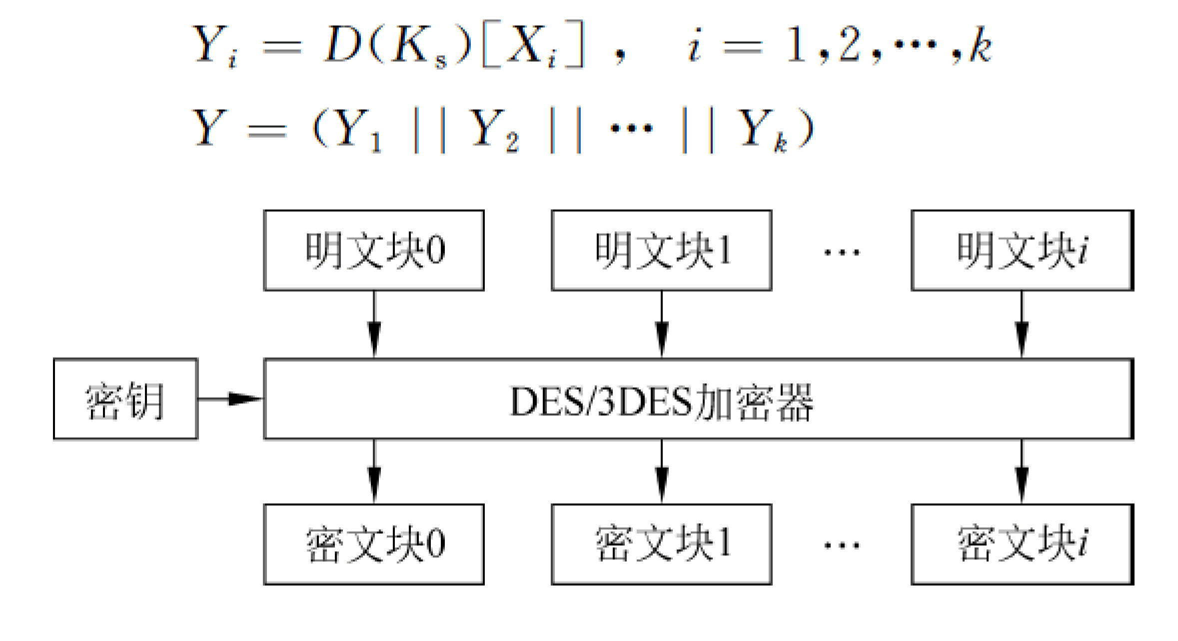


图 4-3　ECB 模式分组加密

2. CBC 模式

在 CBC 模式中，文本块是连续加密的，在加密当前明文块之前，用前一次块加密的结果修改当前明文块。这个过程改进了 ECB 模式的缺点（例如，相同的明文块不会再产生相同的密文块），但是由于其加密过程是连续的，CBC 方式不支持加密的并行化。CBC 方式使用初始向量（initialization vector，IV）用于修改被加密的第一个明文块。

CBC 模式的解密过程描述如下，参见图 4-4。

$$Y_i = E(K_s)[X_i \oplus Y_{i-1}], \quad i = 1, 2, \cdots, k$$

$$Y_0 = (\text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’} \mid\mid \text{‘00’})。$$

$$Y = (Y_1 \mid\mid Y_2 \mid\mid \cdots \mid\mid Y_k)$$

CBC 模式的解密过程

$$Y_i = D(K_s)[X_i] \oplus Y_{i-1}, \quad i = 1, 2, \cdots, k$$

$Y_0 = ('00' \| '00' \| '00' \| '00' \| '00' \| '00' \| '00' \| '00')$。

$Y = (Y_1 \| Y_2 \| \cdots \| Y_k)$

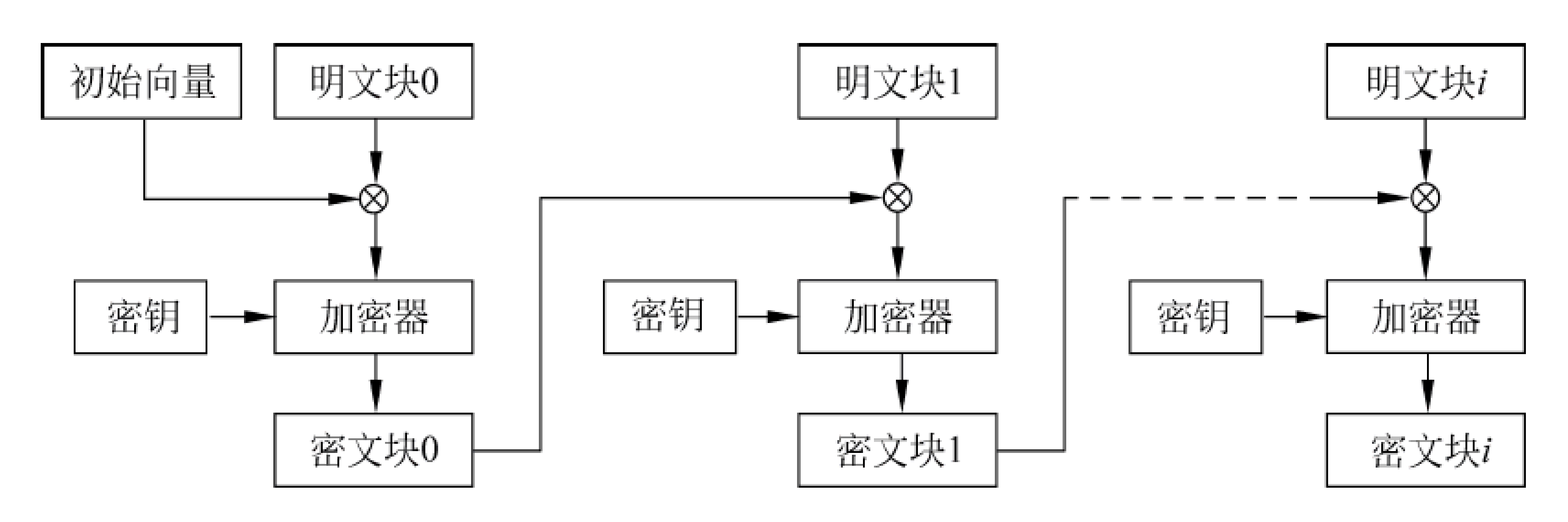


图 4-4 CBC 模式分组加密

4.1.3 数据填充

智能卡进行 DES/3DES 加密时，如果待加密数据的长度不是 8 字节的整数倍，需要进行数据的填充，将数填充到整数块，以便分组加密。

数据填充方式分为多种。在 ISO/IEC 9797-1 规范中给出了 3 种填充方式：

方式 1 在数据尾部填充若干个'00'，将数据填充为 8 字节的整数倍。如果原有数据已经是 8 字节的整数倍，则不再填充。

方式 2 在数据尾部填充一个'80'字节和最少数量个'00'。如果原有数据已经是 8 字节的整数倍，则填充 80 00 00 00 00 00 00 00。

方式 3 在数据尾部填充若干个'00'，将数据填充为 8 字节的整数倍。如果原有数据已经是 8 字节的整数倍，则数据尾部不再填充。并将在数据前端填充若干个'00'和 LD(一个或多个字节表示)。LD 为数据的位长度。

举例如下：

(1) 原始数据块为 8 字节的整数倍。数据块数据如下范例：

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4	AB	61	47

按 ISO/IEC 9797-1 标准的填充方式 1 填充，数据不变。

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4	AB	61	47

按 ISO/IEC 9797-1 标准的填充方式 2，数据尾部添加'80'和 7 个'00'。

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4	AB	61	47
D4	***80***	***00***	***00***	***00***	***00***	***00***	***00***	***00***

按 ISO/IEC 9797-1 标准的填充方式 3，数据前端添加若干个‘00’和长度字节 LD＝24＊8＝192＝‘C0’，尾部不变。

D1	*00*	*00*	*00*	*00*	*00*	*00*	*00*	*C0*
D2	6D	F0	51	3F	1A	DC	1D	6A
D3	1D	2B	7C	87	CB	E1	00	B1
D4	29	44	BF	BB	B4	AB	61	47

（2）原始数据块不为 8 字节的整数倍。数据块数据如下范例：

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4			

按 ISO/IEC 9797-1 标准的填充方式 1，数据尾部填充若干个‘00’。

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4	***00***	***00***	***00***

按 ISO/IEC 9797-1 标准的填充方式 2，数据尾部添加‘80’和若干个‘00’。

D1	6D	F0	51	3F	1A	DC	1D	6A
D2	1D	2B	7C	87	CB	E1	00	B1
D3	29	44	BF	BB	B4	***80***	***00***	***00***

按 ISO/IEC 9797-1 标准的填充方式 3，数据前端添加若干个‘00’和 LD＝21＊8＝168＝‘A8’，尾部填充若干个‘00’。

D1	***00***	***00***	***00***	***00***	***00***	***00***	***00***	***A8***
D2	6D	F0	51	3F	1A	DC	1D	6A
D3	1D	2B	7C	87	CB	E1	00	B1
D4	29	44	BF	BB	B4	00	00	00

其中最常用的填充方式为 ISO/IEC 9797-1 标准的填充方式 2，用一个字节的‘80’和最小数量的字节‘00’对待加密数据进行填充，使新加密数据 NewData 的长度为 8 的整数倍。如果数据 Data 的长度是 8 的整数倍，用一个字节的‘80’和 7 个字节‘00’对 MSG 进行填充。

NewData＝（Data||80||00||…||00）

4.2 AES 算法

1997 年初美国国家标准和技术研究所(NIST)发起并组织了全世界广泛征集新的加密标准算法的活动，这个新加密标准算法被命名为 AES(advanced encryption standard)。NIST 希望 AES 作为一种公开免费的保护敏感信息且全球通用的算法来代替 DES。在征集公告中，NIST 对算法的基本要求是：算法必须是对称密码体制的分组密码，支持 128 位分组长度和 128/192/256 位可选密钥长度。在 AES 征集活动开始后，世界各国许多组织和机构反响强烈。

1998 年 8 月，NIST 在美国加州 Ventura 召开了第一届 AES 候选会议，宣布 CAST-256、Crypton、DEAL、DFC、E2、Frog、HPC、LOK197、Magenta、MARS、RC6、Rijndael、SAFER＋，Serpent、Twofish 这 15 个算法被接受为候选算法。

1999 年 3 月，NIST 在意大利罗马召开第二届 AES 会议，提交对 15 个候选算法的分析结果，讨论它们的安全性、效率和设计灵活性等问题，并确定了入围下一轮评估的 5 个候选算法，分别是 MARS、RC6、Rijndael、Serpent 和 Twofish。随后 NIST 邀请密码专家对这个 5 个候选算法进行了强化攻击，会同世界密码界对各算法的安全性、速度及其通用性等要素进行了进一步评估。

2000 年 4 月，NIST 在纽约召开了第三届 AES 会议，该会提交了对 5 个候选算法的安全性、速度及通用性等要素的分析结果。经过了激烈地讨论，最终由比利时的密码专家 Joan Daemen 和 Vincent Rijmen 所提出的加密算法 Rijndael 以其简洁、高效、安全等优点脱颖而出。2000 年 10 月 2 日 NIST 初步选定 Rijndael 为 AES 算法，随后在公众评议的基础上对草案进行修改，于 2001 年 11 月 26 日发布了正式的 197 号标准——AES 标准，并指出 2002 年 5 月 26 号为标准生效的具体时间。

Rijndael 算法能抵抗目前已知的所有攻击，特别是线性分析和差分分析，也能抵抗某些物理攻击，在和其他算法的安全性相同时它需要的轮数最少。Rijndael 设计简单，不管是硬件上还是低内存环境下它都能快速执行，在所有的平台上它都具有卓越的性能。Rijndael 的密钥方案简单，生成速度快，对内存要求低，非常适合智能卡的实现。

4.2.1 数学基础

有限域是含有有限个元素的域，元素个数称为域的阶。q 阶域存在当且仅当 $q=p^n$，p 为素数并称为有限域的特征，有限域记为 $GF(p^n)$。

Rijndael 算法中涉及两类如下形式的特征 2 域上的多项式

$$b(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0, \quad b_i \in GF(2)$$

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0, \quad a_i \in GF(2^8)$$

其中 $b(x)$ 表示 Rijndael 算法中的单字节数据，$a(x)$ 表示 AES 算法中的 4 字节或字数据。单字节也可以用数据向量 $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$。

单字节运算是两个形如 $b(x)$ 的多项式的运算。单字节运算包括：

(1) 单字节加法。两个多项式的相同指数项系数的模 2 加。

(2) 单字节乘法。两个多项式模 $m(x)=x^8+x^4+x^3+x+1$ 乘。$m(x)$ 为 8 次既约多项式，模 $m(x)$ 能保证相乘后的多项式始终在 $GF(2^8)$ 上。

字运算为两个形如 $a(x)$ 的多项式的运算。字运算包括：

(1) 字加法。两个多项式的相同指数项系数的对应位的模 2 加。例：

$$a(x)=a_3x^3+a_2x^2+a_1x+a_0, A(x)=A_3x^3+A_2x^2+A_1x+A_0$$

则

$$a(x)+A(x)=(a_3 \oplus A_3)x^3+(a_2 \oplus A_2)x^2+(a_1 \oplus A_1)x+(a_0 \oplus A_0)$$

(2) 字乘法。两个多项式模 $M(x)$ 乘。$M(x)=x^4+1$，$M(x)$ 是可约多项式，模 $M(x)$ 可以保证字乘之后的结果仍为一个字。由于 $M(x)$ 不是 $GF(2^8)$ 上的不可约多项式，故并非所有多项式在模 $M(x)$ 下均有乘法逆元。

4.2.2 算法描述

Rijndael 算法是可变分组长和可变密钥长的迭代分组密码，分组长和密钥长是不相关的，可以为 128 位、192 位或 256 位。中间的加密结果称为状态，它可以用一个行数为 4，列数为 N_b 的矩阵来表示，其中元素为字节。种子密钥也可以用一个行数为 4，列数为 N_k 来表示，其中元素为字节。计算的轮数记为 N_r。

N_r 和 N_b、N_k 的关系如表 4-1 所示。

表 4-1 计算轮数 N_r 和状态矩阵列数 N_b，密钥矩阵列数 N_k 的关系

N_r	$N_b=4$	$N_b=6$	$N_b=8$
$N_k=4$	10	12	14
$N_k=6$	12	12	14
$N_k=8$	14	14	14

下面以 128 位分组长度（$N_b=4$）和 128 位密钥（$N_k=4$）为例进行讨论，不影响工作原理的讨论，此时 $N_r=10$。内部状态结构和密钥种子结构如图 4-5 所示。

$$\begin{pmatrix} S_0 & S_1 & S_2 & S_3 \\ S_4 & S_5 & S_6 & S_7 \\ S_8 & S_9 & S_{10} & S_{11} \\ S_{12} & S_{13} & S_{14} & S_{15} \end{pmatrix} \qquad \begin{pmatrix} K_0 & K_1 & K_2 & K_3 \\ K_4 & K_5 & K_6 & K_7 \\ K_8 & K_9 & K_{10} & K_{11} \\ K_{12} & K_{13} & K_{14} & K_{15} \end{pmatrix}$$

图 4-5 内部状态结构和密钥种子结构

轮密钥的生成是由密钥扩展和轮密钥选择两部分构成，其基本原则是：

(1) 轮密钥的总位数等于分组长乘以（$1+N_r$）。

(2) 种子密钥扩展为扩展密钥，种子密钥长度为 $4\times N_k$ 个字节。

(3) 轮密钥由以下方法从扩展密钥中获得。

第 1 轮密钥由前 N_b 个 Int 数（4 字节）构成。

第 2 轮密钥由第二个 N_b 个 Int 数（即第 N_b+1 个字到第 $2N_b$ 个字构成）。

以下依此类推。

举例：对于 128 位分组和 128 位密钥长度的情况，轮密钥总长度为 $16\times(1+10)=176$ 字节；种子密钥长度为 $4\times4=16$ 个字节；第 1 轮的轮密钥为 4 个字节构成，依此类推。

加密过程描述如下，源数据按分组长度分组，每组明文需要经过 N_r 轮的轮变换得到相应密文。对于 128 位分组和 128 位密钥，轮数 $N_r=10$。AES 的加密过程如图 4-6 所示。

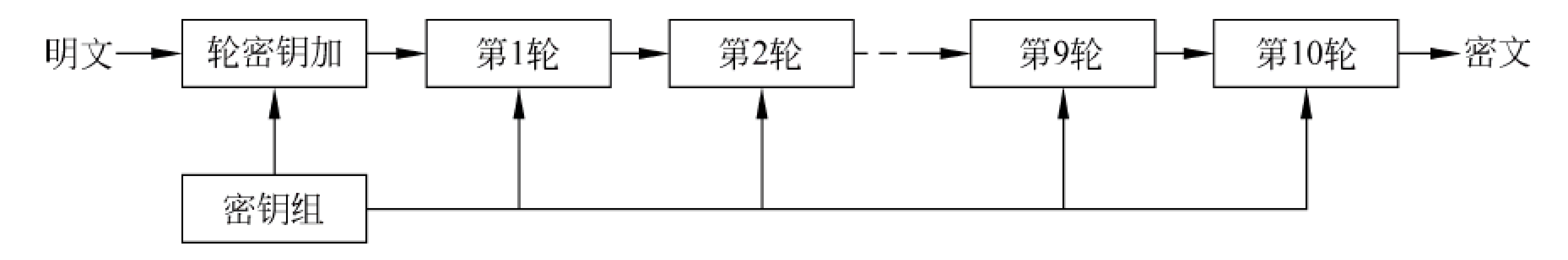


图 4-6 AES 加密过程

其中 $1\sim(N_r-1)$轮的加密轮变换如图 4-7 所示。轮变换由 4 个不同的变换组成，用类 C++ 语言描述为：

```
Round( State, RoundKey )
{
   SubBytes ( State ) ;
   ShiftRows ( State ) ;
   MixColumns ( State );
   AddRoundKey( State, RoundKey );
}
```

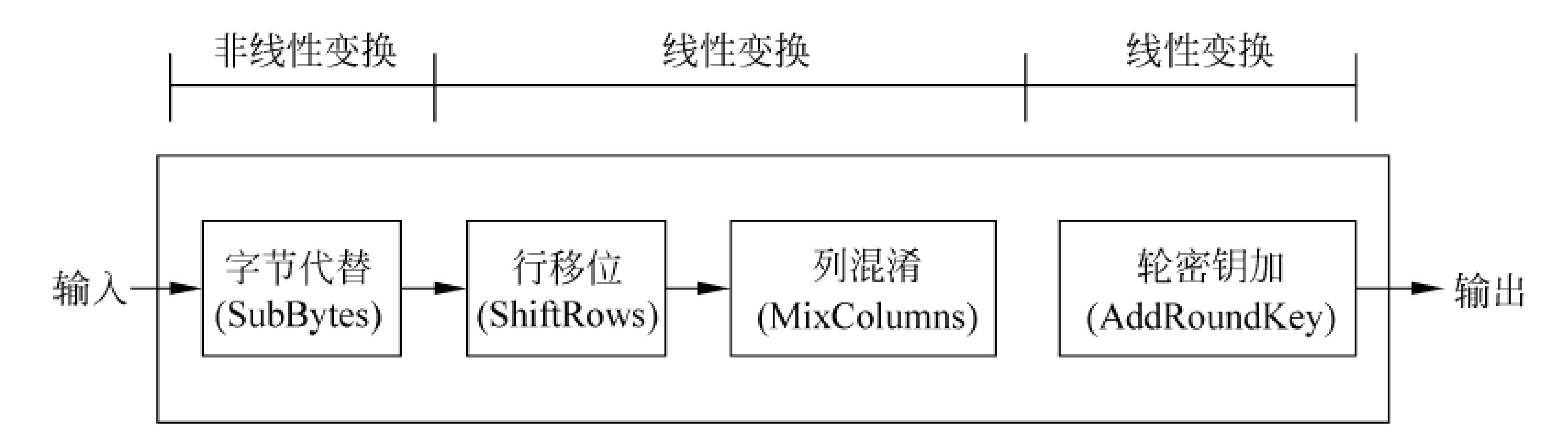


图 4-7　每轮的加密轮变换

而最后一轮稍有不同，删去了 MixColumns(State)函数，即为：

```
FinalRound ( State, RoundKey )
{
    SubBytes ( State ) ;
    ShiftRows ( State ) ;
    AddRoundKey( State, RoundKey );
}
```

下面简要介绍加密轮变换中的各个函数：

(1) SubBytes(State)函数。

状态矩阵的每一字节 x，经过非线性变换公式 $Y=Ax^{-1}+b$ 生成 Y，其中参数如图 4-8 所示。

$$A=\begin{pmatrix}1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\\1&1&1&1&1&0&0&0\\1&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\end{pmatrix}\qquad b=\begin{pmatrix}1\\1\\0\\0\\0\\1\\1\\0\end{pmatrix}$$

图 4-8　状态矩阵参数

如果 x 是零字节，则 $Y=b$。因为 A 是可逆的，所以非线性变换也是可逆的，SubBytes 函数则是可逆的。

Rijndael 算法中的 SubBytes、InvSubBytes 以及密钥扩展中的 SubWord 均为非线性变换。如果采用求字节在 $GF(2^8)$上的逆元，再进行仿射变换的方法来进行以上的非线性变换，速度必然很慢。通常采用预计算查表的方式，构造 $GF(2^8)$上所有元素与经过求逆、仿射变换后的元素之间的映射表即 S 盒，逆 S 盒可采用类似的方法构造。

如：

$$x = 00, \quad x^{-1} = 00, \quad Y = Ax^{-1} + b = 63$$
$$x = 01, \quad x^{-1} = 01, \quad Y = Ax^{-1} + b = 7c$$

…具体值参见表 4-2。

表 4-2 Rijndael 算法中的 S 盒表

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	g5	bc	f6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	e	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6b	8b	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	da	78	25	2e	1c	a6	a4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	65	bb	16

下面举例说明状态矩阵如何转换到下一状态矩阵。

$$\begin{pmatrix} 19 & A0 & 9A & E9 \\ 3D & F4 & C6 & F8 \\ E3 & E2 & 8D & 48 \\ BE & 2B & 2A & 08 \end{pmatrix} \longrightarrow \begin{pmatrix} D4 & E0 & B8 & 1E \\ 27 & BF & B4 & 41 \\ 11 & 98 & 5D & 52 \\ AE & F1 & E5 & 30 \end{pmatrix}$$

第一个字节‘19’，在 S 盒中的 $x=1, y=9$ 处得出下一状态字节为‘D4’。其他字节也采用同样方式得出。

(2) ShiftRows(State)函数。

行移位函数将状态矩阵中的元素进行重排。对于第 i 行的元素，按循环向左移动 i 个位置即可。例如：

$$\begin{pmatrix} D4 & E0 & B8 & 1E \\ 27 & BF & B4 & 41 \\ 11 & 98 & 5D & 52 \\ AE & F1 & E5 & 30 \end{pmatrix} \longrightarrow \begin{pmatrix} D4 & E0 & B8 & 1E \\ BF & B4 & 41 & 27 \\ 3D & 52 & 11 & 98 \\ 30 & AE & F1 & E5 \end{pmatrix}$$

(3) MixColumns(State)函数。

列混淆函数是对状态矩阵中列到列的变换，记做 MixColumns，它作用于状态阵列的每一列，矩阵表示为

$$\begin{pmatrix}S_{0,j}\\S_{1,j}\\S_{2,j}\\S_{3,j}\end{pmatrix}=\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}S_{0,j}\\S_{1,j}\\S_{2,j}\\S_{3,j}\end{pmatrix}$$

例如：

$$\begin{pmatrix}04\\66\\81\\E5\end{pmatrix}=\begin{pmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{pmatrix}\begin{pmatrix}D4\\BF\\5D\\30\end{pmatrix}$$

```
//下面演示第一行的运算过程。
// 1 * state mod x^8 + x^4 + x^3 + x + 1  =  state
static BYTE MulByte01(BYTE b)
{
    return b;
}

// (x) * state mod x^8 + x^4 + x^3 + x + 1
static BYTE MulByte02(BYTE b)
{
    if (b < 0x80)
        return (BYTE)(int)(b << 1);
    else
        return (BYTE)( (int)(b << 1) ^ (int)(0x1b) );
}
// (x + 1 ) *  state mod x^8 + x^4 + x^3 + x + 1 = (x *  state  +  state ) = MulByte02 ( state ) +
MulByte01(state)
static BYTE MulByte03(BYTE b)
{
     return (BYTE) ( (int)MulByte02(b) ^ (int)b );
}
     BYTE X[4] = {0xD4,0xBF,0x5D,0x30};
     BYTE Y[4];
     Y[0] = MulByte02(X[0]);           //B3
     Y[1] = MulByte03(X[1]);           //DA
     Y[2] = MulByte01(X[2]);           //5D
     Y[3] = MulByte01(X[3]);           //30
     BYTE Y1 = Y[0]^Y[1]^Y[2]^Y[3];//结果为 04
////////////////////////////////////////////////////////////////////
```

InvMixColumns 是 MixColumns 函数的逆变换，矩阵表示为

$$\begin{vmatrix}S_{0,j}\\S_{1,j}\\S_{2,j}\\S_{3,j}\end{vmatrix}=\begin{vmatrix}E&B&D&9\\9&E&B&D\\D&9&E&B\\B&D&9&E\end{vmatrix}\begin{vmatrix}S_{0,j}\\S_{1,j}\\S_{2,j}\\S_{3,j}\end{vmatrix}$$

(4) AddRoundKey(State,RoundKey)函数。

将逐字节把状态矩阵和轮密钥矩阵的元素进行异或加,逆函数即自身。

4.2.3 计算范例

详细的 AES 算法见附录 A,举例说明 128 位的 AES 加密计算。

原文 SRC=

52 79 A0 42 46 F0 85 FA 31 3A C5 1E 53 66 C4 D9

3E 93 B5 DE E6 B3 52 50 F7 8F D0 E0 6B 0C 12 35

密钥 Key=

BD 85 95 20 7A ED BF B2 D0 01 80 C2 ED 0C 7C B0

密文 DST=

A2 CE 40 D1 E9 7A FB E9 B0 F5 68 FC 25 00 7D 94

D6 05 14 1D AA E3 8B 53 99 3C 3D 45 B0 24 3A FB

4.3 RSA 算法

1976 年 Diffie 和 Hellman 发表了题为 *New Direction in Cryptography* 的论文,提出了适应网络安全通信的公钥密码思想,揭开了公钥密码研究的序幕。从数学抽象而言,公钥密码体制是一种陷门单向函数,定义域中的任意 x 都容易计算 $f(x)$,而对 f 的值域中的几乎所有 y,都无法计算 $f^{-1}(y)$。只有给出某些辅助信息(陷门信息)时,$f^{-1}(y)$才能顺利计算得出。

1978 年,美国麻省理工学院(MIT)的 Ronald. L. Rivest、Adi. Shamir 和 Leonard. Adleman 共同发表了题为 *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems* 的论文,该文中提出了著名的 RSA 公钥密码体制。RSA 是继 MH 背包公钥密码体制后提出的第一个有效的公钥密码体制。由于算法完善、安全性好、易于实现和理解,RSA 已成为一种应用极广的公钥密码体制。在广泛的应用中,不仅它的实现技术日趋成熟,而且其安全性逐渐得到证明。

在密码学应用中,目前有 3 类系统被公认为是安全且有效的。

(1) 基于整数因式分解难题(简记为 IFP)的公钥密码体制:其中主要包括 RSA 体制和 Rabin 体制。RSA 的安全性依赖于作为公钥的大数 N 的位数长度。为保证足够的安全性,一般认为个人的应用要 512 位的 N,公司需要用 1024 位的 N,极其重要的场合该用 2048 位的 N。随着密钥长度增加,RSA 算法的保密强度增加。但是,密钥越长,其加解密所耗的时间越长,增大模的长度则带来了实现上的难度。

(2) 基于有限域乘法群上的离散对数难题(简记为 DLP)的公钥密码体制:其中主要包括 DSA 签名方案、Diffie-Hellman 密钥交换方案、ElGamal 加密体制和签名体制、Schnorr 签名方案和 Nyberg-Ruppel 签名方案。和 RSA 不同,DSA 有两个安全强度参数:一个是数域的度数,另一个是子群的度数。

(3) 基于椭圆曲线上的离散对数难题(简记为 ECDLP)的公钥密码体制:其中主要包括椭圆曲线型的 Diffe-Hellman 密钥交换方案、椭圆曲线型的 MQV 密钥交换方案、椭圆曲

线型的数字签名算法、椭圆曲线型的 Schnorr 签名方案和 Nyberg-Ruppel 签名方案等。椭圆曲线密码体制只有一个安全强度参数即生成群的度数(或维数)。

4.3.1 算法描述

RSA 算法的数学基础是数论中的欧拉定理,其安全性依赖于大数因数分解的困难性,即:求两个大素数的乘积在计算上是容易的,但要分解两个大素数的积而求出它的素因子是困难的。

欧拉定理:若整数 a 和 N 互素,则 $a^{\varphi(N)} \equiv 1 \bmod N$,其中 $\varphi(N)$是比 N 小但与 N 互素的正整数个数(欧拉数)。

费马定理:若 p 是素数,$(a\ ,p)=1$,则 $a^{p-1} \equiv 1 \bmod p$。

RSA 算法的构造过程如下描述:

(1) 首先任意选取两个大素数 p、q。

(2) 计算乘积 N:$N=p\times q$。

(3) 得到欧拉数:$\varphi(N)=(p-1)(q-1)$。

(4) 任意选择一个与 $\varphi(N)$互素的整数 e 作为加密密钥,e 为公开密钥 K_{pub}。

(5) 根据 e 求出解密密钥 d,并使 d 满足:$de \equiv 1 \bmod \varphi(N)$,$d$ 为私有密钥 K_{pri}。

(6) 加密:$C \equiv M^e \bmod N$;

解密:$M \equiv C^d \bmod N$。

(7) 签名:$H=\text{Hash}(M)$,$S \equiv H^d \bmod N$;

验证:$S^e \bmod N$ 和 $\text{Hash}(M)$是否相等。

从 RSA 算法描述上可以看到,RSA 算法有两个关键部分:

- 大素数 p、q 的选择,这就涉及到素数筛选问题,参见 4.3.2 节。
- 模幂运算,即 $M^e \bmod N$,$C^d \bmod N$ 参见 4.3.3 节。

4.3.2 素数筛选

素数是大于 1、能且只能被 1 和它本身整除的正整数。对于大数 N 而言,小于 N 的素数总数为 $N/\ln N$,素数资源是非常丰富的,故 p 和 q 的选择范围是非常广的,不可能建立所有素数库破解 RSA 算法。

RSA 密码体系安全的关键在于 p 和 q 的选择,然而经密码分析,发现有一类素数存在某些弱特性,根据这些弱特性,攻击者可以通过分解 N 而直接得到有用信息甚至完全破解密码系统。因此在实施的过程中大素数 p 和 q 的选择至关重要,必须满足某些特性,即这些素数必须是强素数(即安全大素数)。

1984 年 Gordon J. 提出了强素数的概念,强素数 p 应当满足以下 4 个条件:

(1) p 是一个很大的随机素数。

(2) $p-1$ 必须有一个很大的素数因子 r。

(3) $p+1$ 必须有一个很大的素数因子。

(4) $r-1$ 也必须有一个大的素数因子。

该定义于 1986 年写入 ISO-DP-9307。

大素数的产生是现代密码学应用中最重要的步骤,产生素数的一般方法可以分为两类。

1. 确定性素数生成方法

确定性素数生成方法指的是其生成的数必然是素数。现在已有许多种确定性素数生成的方法，主要包括基于 Lucas 定理和基于 Pocslington 定理的确定性素数生成方法。

2. 概率性素数生成方法

概率性素数生成方法与确定性素数生成方法的最大不同在于概率性素数生成方法生成的数 p 为素数的概率为 $1-\varepsilon$，其中 ε 为素数生成方法中可控制的任意小数，但不为 0。此类方法是素数生成的主要方法，其中较为著名的算法有 Miller_Rabin 测试法和 Solovay_Strassen 测试法等。

在 RSA 算法实现中，通常采用概率性素数生成方法生成大素数 p、q，由于概率性素数生成方法所生成的是伪素数，所以要通过素数判定，通过了判定以后才可以被 RSA 公钥密码体制所采用。

在实际应用中，Miller_Rabin 测试法比较常用，下面介绍其基本思想。

首先是准备工作。选择待测的大奇数 p，计算 b，2^b 是能整除 $p-1$ 的最大幂数，然后计算 m，使得 $n=1+2^b m$。

然后进入判定循环，其中：

步骤 1：选择一个小于 p 的随机数 a；

步骤 2：设 $j=0$ 且 $z=a^m \bmod p$；

步骤 3：如果 $z=1$ 或 $z=p-1$，那么 p 通过了测试，可能是素数；

步骤 4：如果 $j>0$ 且 $z=1$，那么 p 不是素数；

步骤 5：设 $j=j+1$，

　　如果 $j<b$ 且 $z\neq p-1$，设 $z=z^2 \bmod p$，然后回到第 S4 步。

　　如果 $z=p-1$，那么 p 通过测试，可能是素数；

步骤 6：如果 $j=b$ 且 $z\neq p-1$，那么 p 不是素数。

在实际操作中，可以通过测试 p 对小素数的整除性，减少运算时间。

下面是 Miller_Rabin 算法的具体实现，在实现中引用小素数表 PrimeTable 进行计算优化。

```
////////////////////////////////////////////////////////
#define PRIMETABLENUM 551
//小素数表
const static int PrimeTable[PRIMETABLENUM] =
{   2,  3,   5,   7,   11,   13,   17,   19,   23,   29,   31,  //本行 11 个素数,其他行 10 个
    37,   41,   43,   47,   53,   59,   61,   67,   71,   73,
    79,   83,   89,   97,   101,  103,  107,  109,  113,  127,
    131,  137,  139,  149,  151,  157,  163,  167,  173,  179,
    181,  191,  193,  197,  199,  211,  223,  227,  229,  233,
    239,  241,  251,  257,  263,  269,  271,  277,  281,  283,
    293,  307,  311,  313,  317,  331,  337,  347,  349,  353,
    359,  367,  373,  379,  383,  389,  397,  401,  409,  419,
    421,  431,  433,  439,  443,  449,  457,  461,  463,  467,
    479,  487,  491,  499,  503,  509,  521,  523,  541,  547,
    557,  563,  569,  571,  577,  587,  593,  599,  601,  607,
    613,  617,  619,  631,  641,  643,  647,  653,  659,  661,
```

```
    673,  677,  683,  691,  701,  709,  719,  727,  733,  739,
    743,  751,  757,  761,  769,  773,  787,  797,  809,  811,
    821,  823,  827,  829,  839,  853,  857,  859,  863,  877,
    881,  883,  887,  907,  911,  919,  929,  937,  941,  947,
    953,  967,  971,  977,  983,  991,  997,  1009, 1013, 1019,
    1021, 1031, 1033, 1039, 1049, 1051, 1061, 1063, 1069, 1087,
    1091, 1093, 1097, 1103, 1109, 1117, 1123, 1129, 1151, 1153,
    1163, 1171, 1181, 1187, 1193, 1201, 1213, 1217, 1223, 1229,
    1231, 1237, 1249, 1259, 1277, 1279, 1283, 1289, 1291, 1297,
    1301, 1303, 1307, 1319, 1321, 1327, 1361, 1367, 1373, 1381,
    1399, 1409, 1423, 1427, 1429, 1433, 1439, 1447, 1451, 1453,
    1459, 1471, 1481, 1483, 1487, 1489, 1493, 1499, 1511, 1523,
    1531, 1543, 1549, 1553, 1559, 1567, 1571, 1579, 1583, 1597,
    1601, 1607, 1609, 1613, 1619, 1621, 1627, 1637, 1657, 1663,
    1667, 1669, 1693, 1697, 1699, 1709, 1721, 1723, 1733, 1741,
    1747, 1753, 1759, 1777, 1783, 1787, 1789, 1801, 1811, 1823,
    1831, 1847, 1861, 1867, 1871, 1873, 1877, 1879, 1889, 1901,
    1907, 1913, 1931, 1933, 1949, 1951, 1973, 1979, 1987, 1993,
    1997, 1999, 2003, 2011, 2017, 2027, 2029, 2039, 2053, 2063,
    2069, 2081, 2083, 2087, 2089, 2099, 2111, 2113, 2129, 2131,
    2137, 2141, 2143, 2153, 2161, 2179, 2203, 2207, 2213, 2221,
    2237, 2239, 2243, 2251, 2267, 2269, 2273, 2281, 2287, 2293,
    2297, 2309, 2311, 2333, 2339, 2341, 2347, 2351, 2357, 2371,
    2377, 2381, 2383, 2389, 2393, 2399, 2411, 2417, 2423, 2437,
    2441, 2447, 2459, 2467, 2473, 2477, 2503, 2521, 2531, 2539,
    2543, 2549, 2551, 2557, 2579, 2591, 2593, 2609, 2617, 2621,
    2633, 2647, 2657, 2659, 2663, 2671, 2677, 2683, 2687, 2689,
    2693, 2699, 2707, 2711, 2713, 2719, 2729, 2731, 2741, 2749,
    2753, 2767, 2777, 2789, 2791, 2797, 2801, 2803, 2819, 2833,
    2837, 2843, 2851, 2857, 2861, 2879, 2887, 2897, 2903, 2909,
    2917, 2927, 2939, 2953, 2957, 2963, 2969, 2971, 2999, 3001,
    3011, 3019, 3023, 3037, 3041, 3049, 3061, 3067, 3079, 3083,
    3089, 3109, 3119, 3121, 3137, 3163, 3167, 3169, 3181, 3187,
    3191, 3203, 3209, 3217, 3221, 3229, 3251, 3253, 3257, 3259,
    3271, 3299, 3301, 3307, 3313, 3319, 3323, 3329, 3331, 3343,
    3347, 3359, 3361, 3371, 3373, 3389, 3391, 3407, 3413, 3433,
    3449, 3457, 3461, 3463, 3467, 3469, 3491, 3499, 3511, 3517,
    3527, 3529, 3533, 3539, 3541, 3547, 3557, 3559, 3571, 3581,
    3583, 3593, 3607, 3613, 3617, 3623, 3631, 3637, 3643, 3659,
    3671, 3673, 3677, 3691, 3697, 3701, 3709, 3719, 3727, 3733,
    3739, 3761, 3767, 3769, 3779, 3793, 3797, 3803, 3821, 3823,
    3833, 3847, 3851, 3853, 3863, 3877, 3881, 3889, 3907, 3911,
    3917, 3919, 3923, 3929, 3931, 3943, 3947, 3967, 3989, 4001
};

/*****************************************************************
Miller_Rabin 算法测试素数
返回值：若 *this 为素数,返回 1,否则返回 0
*****************************************************************/
int CHugeNum::IsPrime()
{
```

```
unsigned i,j,pass;
 CHugeNum tmp;

 for(i = 0;i < PRIMETABLENUM;i++)
{
   tmp.Copy(PrimeTable[i]);
   if( Cmp(tmp)  = =  0)           //与小素数相等,返回"素数"
        return 1;
   //对小素数取模,如果余数 = 0,则可整除该素数,则返回"非素数"
   j  =  Mod( PrimeTable[i] ) ;
   if( j = = 0 )
      return 0;
}

CHugeNum S,B,I,K;

 K.Copy( * this);                  //待判断 N 是否是素数
 K.m_ulValue[0] -- ;               //K  =  N  -  1

 for(i = 0;i < 5;i++)              //进行 t( = 5)次循环, 错误概率不会超过 4^ - 5 = 1/1024
 {
     pass = 0;                     //是否通过检测
     B.Copy(rand() * rand());      //取小于 K 的随机数
   S.Copy(K);

     while((S.m_ulValue[0]&1) = = 0)
     //因为 J 属于[0,L - 1],所以形如 11000 一直需要移位到 0 为止
   {
      ///////////////////
      //B^( 2^J  * M )  =  - 1 mod N, 下面就找 J
      //这个循环,则将 S/2
      for(j = 0;j < S.m_nLength;j++)
      {
          S.m_ulValue[j] = S.m_ulValue[j]>> 1;

          if(S.m_ulValue[j + 1]&1)
            S.m_ulValue[j] = S.m_ulValue[j]|0x80000000;
      }

       if(S.m_ulValue[S.m_nLength - 1] = = 0)
         S.m_nLength -- ;
      ///////////////////

      I.Copy( B.ModX(S, * this) ); //S  =  2^J,  I  =  B^( S  * M )mod N
      if(I.Cmp(K) = = 0)           //如果 I  =  N - 1
      {
         pass = 1;
         break;
      }
   }
```

```
            //如果 I = 1
            if((I.m_nLength = = 1)&&(I.m_ulValue[0] = = 1))
               pass = 1;

            //5 次,只要有一次不通过则该数不是素数
            if(pass = = 0)
                 return 0;
         }
         return 1;
      }
      ///////////////////////////////////////////
```

4.3.3 模幂运算

进行幂运算最直接的方法是累乘法,比如计算 x^e,就是先计算 x^2,然后计算 $x^3=x*x^2$,依次计算下去得到 $x^e=x*x^{e-1}$。如果 e 的值很大,使用这样的方法则效率很低。在公钥算法中为了保证加密强度要求,幂运算中所选的指数均很大,因此为保证加密解密速度不能使用累乘法。为了解决幂运算的速度问题,目前几乎所有的快速模幂算法都是基于二进制算法(binary representations,BR)。对二进制算法进行扩展,可以得到效率更高的窗口算法。

模幂运算的基本形式为求解

$$y \equiv x^e \bmod n$$

BR 算法的思想是在计算 x^e 时设 e 二进制化后的长度为 n(其中 $n=\log_2 e$)位,将 e 表示为一个二元整数

$$e = e^0 + e_1 2 + e_2 2^2 + \cdots + e_{n-1} 2^{n-1} = \sum_{i=1}^{n-1} e_i 2^i$$

那么 $x^e=x^{e_0}(x^2)^{e_1}\cdots(x^{2^{n-1}})^{e_{n-1}}$,其中 x 方幂运算可通过计算 x 的平方来完成循环运算。因为 e_i 属于(0,1),所以此算法需要做约 n 次乘法运算加 n 次平方运算。该算法在进行实际计算时由于计算的最终目的均是 $x^e \bmod m$,为控制中间结果数的大小,为此每做一次乘法运算后还应做一次模运算。

在使用 BR 算法进行 RSA 模幂运算时,考虑到参数 $N=p*q$ 的关系,可采用中国剩余定理简化模幂运算中的幂次数。

我国最早的同余问题出现在数学著作《孙子算经》中,其中记载了“物不知其数”问题:“今有物,不知其数,三三数之剩二;五五数之剩三;七七数之剩二,问物几何?”这个问题相当于求解同余方程组

$$\begin{cases} x \equiv 2 \pmod 3 \\ x \equiv 3 \pmod 5 \\ x \equiv 2 \pmod 7 \end{cases}$$

这个同余方程组的公共解就是问题的答案。

南宋数学家秦九韶在他的著作《数书九章》中提出了解一次同余式组的一般方法。我们今天称之为“中国剩余定理(CRT)”。

中国剩余定理：设 $M_1, M_2, \cdots, M_n$ 是 n 个两两互素的正整数，$r_1, r_2, \cdots, r_n$ 为 n 个给定的整数，那么一次同余方程组

$$x \equiv r_i (\bmod\ M_i), \quad i = 1, \cdots, n$$

必有解，且解数唯一，这个同余方程组的最小正整数解为

$$x = \left(\sum_{i=1}^{n} r_i k_i \frac{M}{M_i} \right) - CM \quad (\text{其中 } C \text{ 为整数}, M = M_1 M_2 \cdots M_n)$$

中国剩余定理的推论：设 $p_1, p_2, \cdots, p_n$ 是 n 个两两互素的正整数，$p = p_1 p_2 \cdots p_n$，则同余式 $f(x) \equiv 0 (\bmod\ p)$ 与 $f(x) \equiv 0 (\bmod\ p_i)$ 等价。

费马小定理：设 p 是一个素数，x 是一个满足 $x (\bmod\ p) \neq 0$ 的整数，则 $x^{p-1} \equiv 1 (\bmod\ p)$。

由于在实际应用中，公开密钥 e 一般选择比较小，通常选择 $e = 2^{16} + 1 = 65537 = (0001\ 0001)_{hex}$，而私有密钥 d 则接近 N 的长度，这样加密/签名验证操作比较快，而解密/签名操作则比较慢。运用中国剩余定理改进 RSA 的私钥运算 $C = M^d \bmod N$，则能够有效地提高 RSA 的运算速度。

在执行解密和签名操作时，大素数 p、q 是已知的，而 $N = p * q$，故

$$C = M^d \bmod N$$

利用中国剩余定理的推论将模 N 运算转化成模 p 和模 q 的运算，即

$$C_p \equiv m^d (\bmod\ p)$$

$$C_q \equiv m^d (\bmod\ q)$$

根据费马小定理，简化方程组

$$dp \equiv d (\bmod\ p - 1)$$

$$dq \equiv d (\bmod\ q - 1)$$

$$C_p \equiv M^{dp} (\bmod\ p)$$

$$C_q \equiv M^{dq} (\bmod\ q)$$

整理上述转化公式

$$C \equiv M^d \bmod N \equiv (C_p (q^{-1} \bmod p) q + C_q (p^{-1} \bmod q) p) \bmod N$$

在这个公式中，将 dp、dq、$iq = (q^{-1} \bmod p)$、$ip = (p^{-1} \bmod q)$ 提前计算好，那么整个计算过程约需要 $3k^3/8$ 次位运算，其中 k 为 N 的二进制长度，如果不用中国剩余定理，则约需要 $3k^3/2$ 次位运算。采用中国剩余定理的 RSA 系统的运算速度提高约 4 倍。

在智能卡应用中，通常用 RSA_CRT 和 RSA_SF(straight forward)区分这两种计算。而 RSA_CRT 通常又分为两种方式：单基数转换法(single-radix conversion，SRC)和混合基数转换法(mixed-radix conversion，MRC)：

(1) SRC 转换法

$$C \equiv M^d \bmod N \equiv (C_p (q^{-1} \bmod p) q + C_q (p^{-1} \bmod q) p) \bmod N$$

需要计算 dp、dq、ip、iq。

(2) MRC 转换法

$$\begin{aligned} C &\equiv M^d \bmod N \equiv (C_p (q^{-1} \bmod p) q + C_q (p^{-1} \bmod q) p) \bmod N \\ &\equiv C_p + [(C_q - C_p) * (p^{-1} \bmod q) (\bmod\ q)] * p \end{aligned}$$

这样只需提前计算好 dp、dq、ip 即可，并省去了最后模 N 操作。不再存储 ip 节省智能卡存储空间的使用。

4.3.4 计算范例

下面列举一个 1024 位 RSA 运算过程的范例，所有数据均高位在前（即 Endian 为大端模式），数据描述为十六进制。

生成 $p=$

35C4D5E28630A070276EB6F63DC414B6 94544AEE9C90959E7904F6064EC6616E
2B34DAF4EA14F40852FECF4CBB0C7E86 7A988C521532D748AC88B44EDD60BA9B

生成 $q=$

9730FA7C9E0ADD465672827AEFA033EE F974A4A05FF42D328144011A12F2FDA6
C022168E1A6E5232C40E6256C200CC6E C9EAE7343F5651B691069290DF0CE48B

计算 $N=p*q=$

1FC163AD1D086CB094CF67200C1F5BC9 CB9541F046C966928A36230AFBDCFAB0
F1160A12ECE0F6E91F60F1247537A20B AB0F3CD77895C16CDE8511B645B516B4
A9F6A75098F3D237A7A2B668790DBD17 515D9A427D6E491A23C7FB3A85472D1C
68CBB323F269B942F0020C8BAB1A92D8 0CD7ACFCEBE3DF5E70D124421DFB5E29

计算 $\varphi(N)=(p-1)(q-1)=$

1FC163AD1D086CB094CF67200C1F5BC9 CB9541F046C966928A36230AFBDCFAB0
F1160A12ECE0F6E91F60F1247537A20B AB0F3CD77895C16CDE8511B645B516B3
DD00D6F174B8548129C17CF74BA97471 C394AAB380E98649297F041A238DCE07
7D74C1A0EDE67307D8F4DAE82E0D47E2 C8543976975AB65F3341DD62618DBF04

选定 e=01 00 01

计算 $d=$

0124DD6B6A25588D0FFF5CCEB9C11374 74A21EEB0211D561281B852DAB1F2DE4
DBFA984850425682D57F8F9069ACA468 D933C65458E2C44220CE18724196A187
BF497D74DB689C9ABE9F5ADA29DEA3F4 49407CD9D4A6151163AE50BE8453627D
C09649377BFA96604CFF9D374A196E62 8C27049EF21AF181027EB7F40861ABE5

计算 $dp \equiv d(\bmod\ p-1)$，$e*dp \bmod (p-1)=1$，

$dp=$

2F84D0BDCC78272257471A7D4EA2FEE7 48F9A1EB7504B6A602302739F5DE8EF6
AD4A9E037AE89FAEC7A2981428EEE6D6 F47BE2EF1BFAFD96259CFF34FFA2DFE7

计算 $dp \equiv d(\bmod\ p-1)$，$e*dp \bmod (p-1)=1$，

$dq=$

166852E77F40DAF49EC14FE74A7672DB FB2AB76F71692DFDAB3CD9F0F4580111
E2ADEAABDAECF8CFBF50E67037468905 EE751B191A3DFFD450002A5EC04017EB

计算 $dp \equiv q^{-1} \bmod p$，$iq*p \bmod p=1$，

$iq=$

321FC75DFC229DC7C6C4D7E83BF34211 2047C0CB1634F09C68841BFAFF74516F
25D83AA98AEABA0C69E9D4C6DA8934E6 68A7B12E53E6FDB2DD5A23C6EEFFE9A5

计算 $ip \equiv q^{-1} \bmod p$，$ip*p \bmod p=1$，

ip=

0A3FA3395D9A0C22F28999D4DBD23C46 CB229E260794D3CCC53137EB727BE6BA
DA6FDB5E8E031C44198B2ADA29B0948D 613E0827D3EF24E356D056CAF7864C69

输入 M=12 34 56 78 AA BB CC DD

计算 $C = M^e \bmod N =$

12CDB0BFB9759F598B78604AD10BC24F 95AB97A35BF3FC4D49AF3677A6E10374
ECECA3556A79F4D1286CE0D6A8B2B9CF BF5A9CC4B24CAD351FD9E9FEB480E170
6D9EEBA9632CFC819968565B7FBCE66D E370E5AF6CD2B48F3FF23BBC5C168B2D
7BD852E23357D0763F2057B331B47DB7 59DD084AD280BF337859A52D818F8859

4.4 ECC 算法

1976 年 Diffie 和 Hellman 提出公钥密码体制时已经使用了有限域上的离散对数问题。椭圆曲线密码 ECC(elliptic curve cryptography)是 1985 年 V. Miller 和 N. Koblitz 各自独立地提出的。ECC 体制的安全性是基于椭圆曲线上离散对数问题求解的困难性,目前还没有找到解决此问题的次(亚)指数时间算法,因而它具有一些其他公钥密码体制无法比拟的优点:

(1) 安全性能更高。加密算法的安全性能一般通过该算法的抗攻击强度来反映。ECC 和其他几种公钥系统相比,其抗攻击性具有绝对的优势。如 160 位的 ECC 与 1024 位的 RSA 有相同的安全强度。而 224 位的 ECC 则与 2048 位的 RSA 具有相同的安全强度。

(2) 计算量小,处理速度快。虽然在 RSA 中可以通过选取较小的公钥的方法提高公钥处理速度,即提高加密和签名验证的速度,使其在加密和签名验证速度上与 ECC 有可比性,但在私钥的处理速度上(解密和签名),ECC 远比 RSA 快得多(如 160 位的 ECC 同 1024 位的 RSA 相比,处理速度相差达几十倍)。

(3) 存储空间占用小。在相同安全强度下,ECC 的密钥尺寸和系统参数与 RSA 相比要小得多,意味着它所占的存储空间要小得多。这对于智能卡应用而言具有特别重要的意义。

4.4.1 数学基础

由代数曲线的 Riemman_Roch 定理可知,任意一条椭圆曲线总可以用一个维尔斯特拉斯方程来表示。设 F 是一个域,则在仿射坐标系下三次方程称为域 F 上的维尔斯特拉斯方程

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

其中 $a_1, a_2, a_3, a_4, a_6 \in F$,该方程在仿射平面 $A^2(\bar{F})$ 中的所有解加上一个无穷远点 O 组成的集合 E 称为域 F 上的一条椭圆曲线,记作 E/F,F 称做椭圆曲线的基域。记作:

$$E/F\{(x,y) \in A^2(\bar{F})\} \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\} \cup \{O\}$$

根据域 F 的不同特征值 $Char(F)$,可以将椭圆曲线分为几类,并且通过适当的代数变换,可将椭圆曲线方程变换成以下简单的形式:

(1) $Char(F)=2$:E:$y^2+y=x^3+ax+b$。

(2) $Char(F)=3$:E:$y^2=x^3+ax^2+bx+c$。

(3) $Char(F)>3$:E:$y^2=x^3+ax+b$。

从便于实现角度考虑，目前对椭圆曲线算法的研究集中在(1)、(3)两类上，即是在二元扩域 $F(2^m)$和素数域 $F(q)$上，$F(q)$根据 q 的不同情况又可分为以下几种：

(1) $q=p$，且为一大素数。

(2) p 为一小素数，$q=p^n$，考虑扩域 $F(p^n)$上的椭圆曲线 $E/F(p^n)$。

(3) p 是大小接近使用该椭圆曲线密码体制的 CPU 一次所能处理的最大字长的一个素数，考虑扩域 $F(p^n)$上的椭圆曲线 $E/F(p^n)$。

在基域 F 上的椭圆曲线上的点的个数也称椭圆曲线的阶，记作 $\#E/F(q)$，$\#E/F(q)$越大，其上的密码系统越安全，通常要求 $\#E/F(q)$包含一个大素数因子或本身就是一个大素数。

4.4.2 运算定义

在椭圆曲线点集合上可以定义适当的加法规则(以“+”表示)使得椭圆曲线 $E/F(q)$中所有点按点的加法规则组成一个有限 Abel 群。即椭圆曲线上任意两点 P、Q 满足：

(1) 封闭性

$$\forall P,Q \in E/F(q),\quad R = P + Q \in E/F(q)$$

(2) 结合律

$$\forall P,Q,S \in E/F(q)\quad (P+Q)+S = P+(Q+S)$$

(3) 单位元存在

$$\forall P \in E/F(q),\quad P+O = O+P = P$$

(4) 任意一个元素都有逆元存在

$$\forall P \in E/F(q),\quad \exists Q \in E/F(q),\quad \text{满足 } P+Q = O$$

(5) 交换律

$$\forall P,Q \in E/F(q),\quad P+Q = Q+P$$

因此，在有限域椭圆曲线 Abel 点群上，也有类似于有限域乘法群上的离散对数问题。例如：设 P 是 $E/F(q)$上的一个点，存在整数 $k\in[1,\#E/F(q)]$，使 $Q=kP$，则椭圆曲线离散对数问题就是由给定的 P,Q 求出 k，这也就是椭圆曲线密码体制的基础。

在不同的坐标系下，椭圆曲线方程的表达式不同，点的表达式不同，点加和倍点的表达式也不同。因而不同的坐标系下计算点加和倍点的计算量不同，选择适当的坐标系是提高椭圆曲线密码算法效率的一个重要途径。

由定义可得 $P+Q$，对于一般的两点 $P,Q\in E(P\neq Q)$，设通过 P、Q 的直线交曲线 E 与另外的一点 T，则 P、Q 和 T 是 E 上的三点，且在同一直线上，也就是说 P 与 Q 的和是过 P 和 Q 的直线与曲线 E 的第三个交点关于 x 轴的对称点。现设点 T 关于 x 轴的对称点 R(如图 4-9 所示)，则 E 中的另外三点 R、T 和 O 也共线。

若 $P\neq Q$ 且 P 和 Q 的 x 坐标相同(y 坐标比定相反)则 $P+Q=O$。

若 $P=Q$，令 L 为椭圆曲线上 P 点的切线，L 截椭圆曲线仅有一点 R，则 $P+Q=2P=R$，由此可以计算 $2P$。

$P+Q$ 称做点加，k 个相同的点相加，即 $P+P+\cdots P$ 表示 kP，则称为点乘或数乘。

在加解密过程中，需要将椭圆曲线离散化，如 $y^2=x^3+x+1 \bmod 13$ 椭圆曲线如图 4-10 所示。

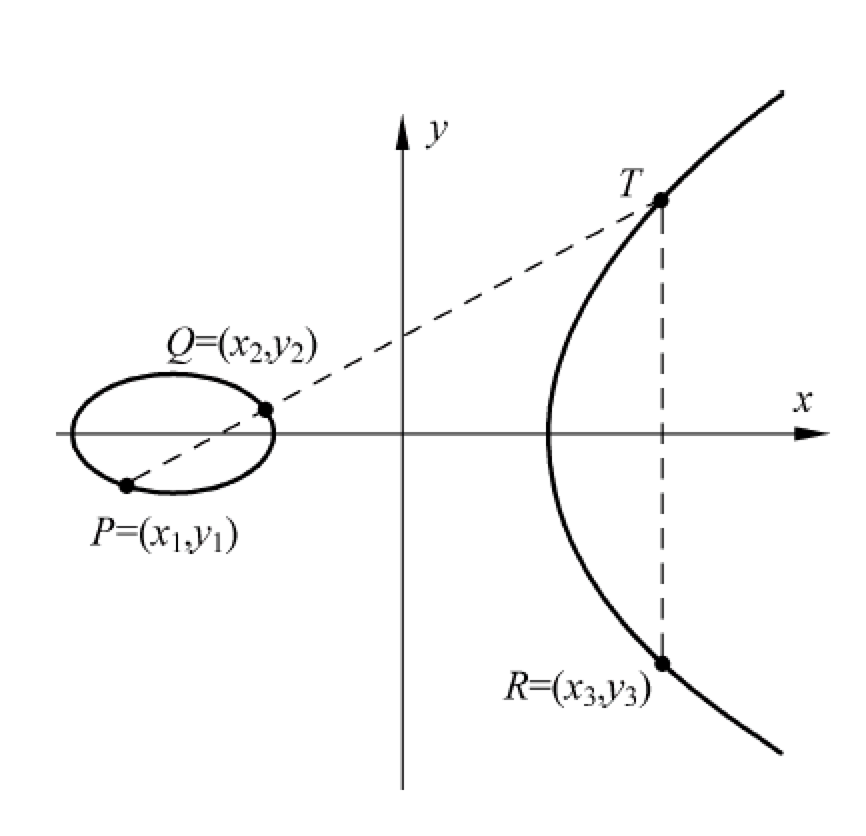


图 4-9 椭圆曲线的加法

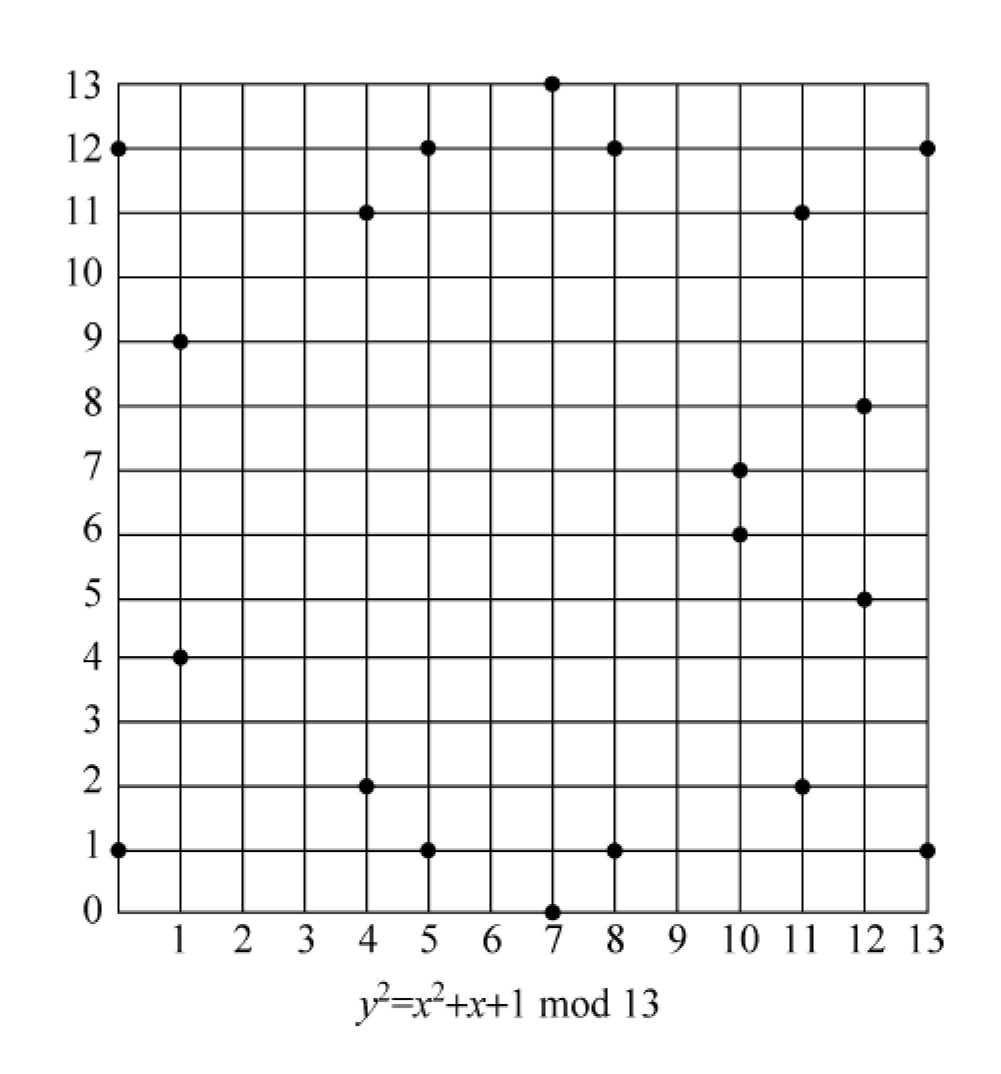


图 4-10 离散椭圆曲线

ECC 的算法属于以精确计算为特征的符号计算范畴，是典型的计算机代数类算法。从点群的角度看，ECC 要用到点加（两个不同点相加）、倍点（两个相同的点相加）、点乘（多个相同的点相加，是倍点和点加的复合），这些运算都要转化成点坐标的计算。椭圆曲线中最基本最重要的运算之一就是标量乘法，即求点 G 的 k 倍。在密码体制的实现中，它的运算速度直接影响到整体速度。目前计算标量乘的算法主要有二元展开法、k 进制方法、滑动窗口法等。

最常用到基于椭圆曲线算法的是数字签名和密钥交换算法，虽然同样是基于椭圆曲线密码学，但是密码算法的形式很多，总体要求是要安全、简单便于实现、功能全面，因此系统的设计者往往会选择最适合自己的协议形式。以下给出两种算法的常见形式：ECDSA 数字签名/验证算法和 Diffie-Hellman 型密钥共享算法。

4.4.3 ECDSA 算法

在 SEC（Standards for Efficient Cryptography）制定的 ECC 工作草案中，定义椭圆曲线参数的形式是一个六元偶，记作

$$R = (q, A, B, G, n, h)$$

其中：整数 q 表示椭圆曲线基域 $F(p^m)$ 中域元素的个数；A，B 表示椭圆曲线方程的系数；G 是椭圆曲线点群的一个基点；n 是椭圆曲线基点 G 的阶；h 是余因子，利用 h 可以较快地找到基点 G。

ECDSA（elliptic curve digital signature algorithm）是不带有消息恢复功能的签名方案。需要使用的参数包括：椭圆曲线参数六元偶；待签名消息 m；一对公私密钥对 d 和 Q，$Q=dG$，其中 d 是私钥，Q 是公钥。

签名过程如下：

（1）选取整数 $k(1<k<n-2)$，计算点 $(x_T, y_T)=kG$，并设 $r = x_T \bmod n$；

（2）利用 Alice 的私钥 d，计算 $s=k^{-1}(m+dr) \bmod n$；

(3) A 送给 Bob 消息 m 及签名(r, s)。

Bob 收到 Alice 的签名(r, s)后，利用 Alice 的公钥 Q 和系统参数，验证过程如下：

(1) 计算 $w = s^{-1} \bmod n$；

(2) 计算 $u_1 = mw \bmod n$ 和 $u_2 = rw \bmod n$；

(3) 计算点$(x_D, y_D) = u_1 G + u_2 Q$，当 $x_D \bmod n = r$ 时，Bob 接受 Alice 对消息 m 的签名。

4.4.4 ECDH 算法

Alice 将要发送消息给 Bob，二者首先协商一组椭圆曲线参数六元偶，然后 Alice 和 Bob 要做的是：

- Alice 随机选取整数 $1 < k_A < n-2$，计算 $Q_A = k_A G$，并将 Q_A 发送给 Bob；
- Bob 随机选取整数 $1 < k_B < n-2$，计算 $Q_B = k_B G$，并将 Q_B 发送给 Alice；
- Alice 收到 Bob 发来的 Q_B 后计算 $k_A Q_B = k_A k_B G$；
- Bob 收到 Alice 发来的 Q_A 后计算 $k_B Q_A = k_B k_A G$。

由于 $Q = k_A k_B G = k_B k_A G$，于是 Alice 和 Bob 共同拥有的会话密钥为 Q。

举例：在扩展访问控制 EAC 机制中，EF. DG14 存储 ECC 曲线参数，通常格式如图 4-11 所示。

```
01 26 30 82 01 22 06 09 04 00 7f 00 07 02 02 01   OID=0.4.0.127.0.7.2.2.1.2
02 30 82 01 13 30 81 d4 06 07 2a 86 48 ce 3d 02   ECDH
01 30 81 c8 02 01 01 30 28 06 07 2a 86 48 ce 3d   OID=1.2.840.10045.1.1
01 01 02 1d 00 d7 c1 34 aa 26 43 66 86 2a 18 30   素数 p
25 75 d1 d7 87 b0 9f 07 57 97 da 89 f5 7e c8 c0
ff 30 3c 04 1c 68 a5 e6 2c a9 ce 6c 1c 29 98 03   参数 a
a6 c1 53 0b 51 4e 18 2a d8 b0 04 2a 59 ca d2 9f
43 04 1c 25 80 f6 3c cf e4 41 38 87 07 13 b1 a9   参数 b
23 69 e3 3e 21 35 d2 66 db b3 72 38 6c 40 0b 04
39 04 0d 90 29 ad 2c 7e 5c f4 34 08 23 b2 a8 7d   参数 G
c6 8c 9e 4c e3 17 4c 1e 6e fd ee 12 c0 7d 58 aa
56 f7 72 c0 72 6f 24 c6 b8 9e 4e cd ac 24 35 4b
9e 99 ca a3 f6 d3 76 14 02 cd 02 1d 00 d7 c1 34   参数 n
aa 26 43 66 86 2a 18 30 25 75 d0 fb 98 d1 16 bc
4b 6d de bc a3 a5 a7 93 9f 02 01 01 03 3a 00 04
b3 c2 1a 5f ee 96 32 49 1e 8e da 4d 66 ea 5a f5
29 64 80 99 3d eb 9d cd 58 63 39 fd 0a f7 60 e7   公钥 Q 的坐标 x，y
70 ac 41 fe ef d6 84 38 87 18 fe 07 aa 47 c0 ae
07 cd f4 b3 b0 08 45 1e
```

图 4-11 ECC 曲线参数

```
//产生 ECDH 密钥对
//PCD 的私钥 dPCD
   48 d2 c6 f0 cc 66 59 14 5d 0c 8c ec 99 7a 99 16
   da e9 dd c1 de 37 b6 05 17 98 cc b8
//PCD 的公钥 QPCD = dPCD * G
   04 80 34 f2 a3 30 a5 f1 ea 11 82 61 b1 5d 1b 6d
   20 60 2f 15 3e e6 f0 b1 97 31 b5 3f d6 57 e7 be
   d1 2e c3 61 fb 19 35 a5 45 16 62 73 66 95 f2 aa
```

```
   e2 b4 b1 b9 cf d7 c7 13 dd
//ECDH shared secret
//K = dPICC * dPCD * G = dPICC * QPCD = dPCD * QPICC
//此时用 PCD 的私钥 * PICC 的公钥,取 x 坐标
   73 21 b8 7a bb 53 d2 4c 7e f2 d4 2a 56 54 ba 9d
   26 53 16 0a b4 22 58 a4 8c 1f bc 94
```

4.4.5 计算范例

下面给出 192 位素数域椭圆曲线的密钥对产生结果。其中方程为

$$y^2 = x^3 + a * x + b (\text{mod } p)$$

$T=(p\ a\ b\ G\ n\ h)$参数为

P = FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE FF FF FF FF FF FF FF FF

a = FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE FF FF FF FF FF FF FF FC

b = 64 21 05 19 E5 9C 80 E7 0F A7 E9 AB 72 24 30 49 FE B8 DE EC C1 46 B9 B1

G. x = 18 8D A8 0E B0 30 90 F6 7C BF 20 EB 43 A1 88 00 F4 FF 0A FD 82 FF 10 12

G. y = 07 19 2B 95 FF C8 DA 78 63 10 11 ED 6B 24 CD D5 73 F9 77 A1 1E 79 48 11

Order =FF FF FF FF FF FF FF FF FF FF FF FF 99 DE F8 36 14 6B C9 B1 B4 D2 28 31

取私钥 k 为随机数

k = 0B 18 21 D1 4E FB 8 C 48 C9 97 0A 6D 90 3B 49 78 3E 81 20 97 10 4C 6F 7F

最后得出结果,公钥 $K=k*G$ 为

K. x= BA 65 67 DA 0F 65 7D 95 BE B6 B2 05 36 9F 30 9A 0C 0B 83 0C 23 B9 42 E6

K. y= A7 90 04 D6 61 F7 4D 2D FE B2 A8 1E 20 A9 BC 99 6F 7D 7A 14 B6 A6 88 82

4.5 SHAx 算法

安全散列标准 SHA(secure hash standard)算法于 1993 年发布,版本为 FIPS PUB 180。这个版本现在常被称为"SHA0"。它在发布之后很快就被美国国家安全局(national security agency,NSA)撤回,并且以 1995 年发布的修订版本 FIPS PUB 180-1(通常称为 SHA1)取代。根据 NSA 的说法,它修正了一个在原始算法中会降低密码安全性的错误。然而 NSA 并没有提供任何进一步的解释或证明该错误已被修正。而后美国国家标准与技术研究院(national institute of standards and technology,NIST)又发布了三个额外的 SHA 变体,每个都有更长的消息摘要。以它们的摘要长度加在原名后面来命名:SHA256、SHA384 和 SHA512。它们发布于 2001 年的 FIPS PUB 180-2 草稿中,随即通过审查和评估。包含 SHA1 的 FIPS PUB 180-2,于 2002 年以官方标准发布。2004 年 2 月,发布了一次 FIPS PUB 180-2 的变更通知,加入了一个额外的变种 SHA-224。这样就形成 5 种安全 SHAx 标准:分别为 SHA1、SHA224、SHA256、SHA384、SHA512。表 4-3 对 SHAx 算法从最大信息长度、分组大小、计算轮数等方面进行了详细比较。

表 4-3 SHAx 算法比较

特　　征	SHA1	SHA224	SHA256	SHA384	SHA512
最大信息长度(位)	$2^{64}-1$	$2^{64}-1$	$2^{64}-1$	$2^{128}-1$	$2^{128}-1$
分组大小(位)	512	512	512	1024	1024
计算轮数	80	64	64	80	80
字节大小	32	32	32	64	64
信息摘要大小(位)	160	224	256	384	512

4.5.1 算法描述

SHA1 对长度不超过 2^{64} 二进制位的消息产生 160 位的消息摘要输出，计算过程如图 4-12 所示。

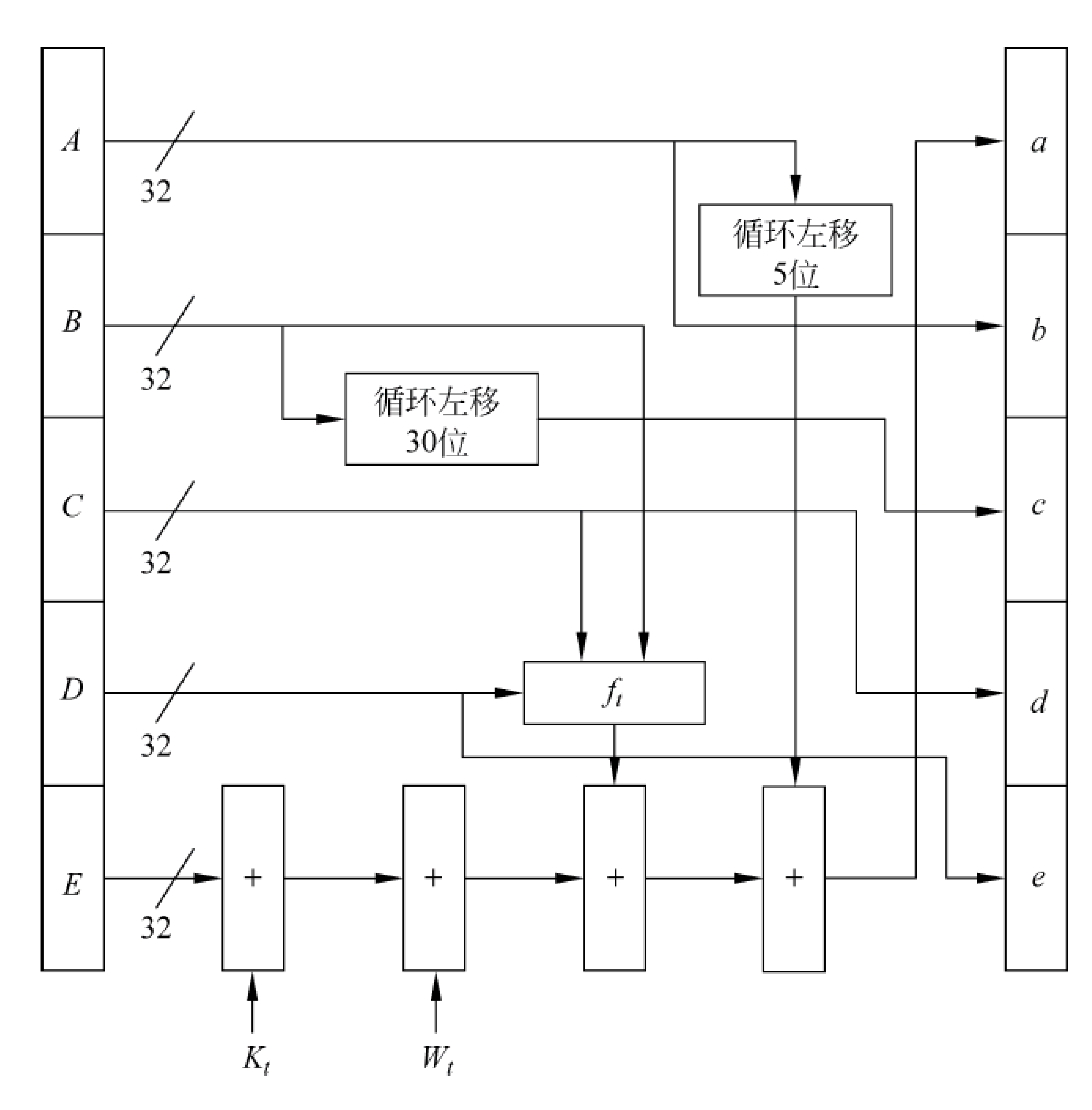


图 4-12 SHA1 计算过程

步骤如下：

(1) 填充消息使其长度恰好为一个比 512 的倍数仅小 64 位的数，消息后填充数据为位串 1—0—0—…—0，然后在其后附上 64 位的消息长度(填充前的长度)，使消息长度恰好是 512 位的整数倍。

(2) 初始化 5 个 32 位变量，用十六进制表示如下：A=0x67452301；B=0xEFCDAB89；C=0x98BADCFE；D=0x10325476；E=0xC3D2E1F0。

(3) 开始算法的主循环，一次处理 512 位消息，循环次数是消息中 512 位分组的数目。先把这 5 个变量复制到另外的变量中，A 到 a，B 到 b，C 到 c，D 到 d，E 到 e。主循环有 4 轮，每轮 20 次操作，每次操作对 a、b、c、d、e 中的 3 个进行一次非线性运算，后进行移位和加

运算,运算的过程见图 4-12。其中 f_t 为逻辑函数,W_t 为由当前 512 位输入分组导出的一个 32 位字,K_t 为额外常数,4 个步骤使用不同的值。K_t 的定义如下:K_1=0x5A827999;K_2=0x6ED9EBA1;K_3=0x8F1BBCDC;K_4=0xCA62CAD6。

(4) a、b、c、d 和 e 分别加上 A、B、C、D 和 E,然后用下一数据分组继续运行算法。

(5) 最后的输出由 A、B、C、D 和 E 级联而成。

4.5.2 算法实现

根据 SHA1 算法描述,采用 C++语言实现如下:

```
CSHA1 sha;
sha.SHA1Reset();
sha.SHA1Input((const unsigned char * ) fileContent, fileLen);
sha.SHA1Result();
```

在 SHA1Input 函数中分组调用 SHA1ProcessMessageBlock 函数,该函数过程实现过程如图 4-12 所示。

```
void CSHA1::SHA1ProcessMessageBlock( )
{
    const unsigned K[] =                          // SHA1 标准中定义的 K 常数
    { 0x5A827999,0x6ED9EBA1,0x8F1BBCDC,0xCA62C1D6 };
    int t;                                        // 循环计数器
    unsigned temp;                                // 临时变量
    unsigned W[80];                               // 导出的字
    unsigned A, B, C, D, E;                       // 32 位的变量

    //初始化 W 数组的前 16 个元素
    for(t = 0; t < 16; t++)
    {
       W[t] = ((unsigned) Message_Block[t * 4]) << 24;
       W[t] |= ((unsigned) Message_Block[t * 4 + 1]) << 16;
       W[t] |= ((unsigned) Message_Block[t * 4 + 2]) << 8;
       W[t] |= ((unsigned) Message_Block[t * 4 + 3]);
    }
    //计算后 64 个 W 数组元素的值
    for(t = 16; t < 80; t++)
    {
       W[t] = SHA1CircularShift(1,W[t - 3] ^ W[t - 8] ^ W[t - 14] ^ W[t - 16]);
    }
    //初始化 5 个 32 位变量
    A = Message_Digest[0];
    B = Message_Digest[1];
    C = Message_Digest[2];
    D = Message_Digest[3];
    E = Message_Digest[4];

//第一轮操作,操作过程见图 4 - 12
    for(t = 0; t < 20; t++)
    {
```

```
        //循环左移 5 位操作
        temp = SHA1CircularShift(5,A) + ((B & C) | ((~B) & D)) + E + W[t] + K[0];
        temp &= 0xFFFFFFFF;
        E = D;
        D = C;
        //循环左移 30 位
        C = SHA1CircularShift(30,B);
        B = A;
        A = temp;
    }

//第二轮操作,操作过程见图 4-12
    for(t = 20; t < 40; t++)
    {
        //循环左移 5 位操作
        temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[t] + K[1];
        temp &= 0xFFFFFFFF;
        E = D;
        D = C;
        //循环左移 30 位
        C = SHA1CircularShift(30,B);
        B = A;
        A = temp;
    }

//第三轮操作,操作过程见图 4-12
    for(t = 40; t < 60; t++)
    {
        //循环左移 5 位操作
temp = SHA1CircularShift(5,A) +   ((B & C) | (B & D) | (C & D)) + E + W[t] + K[2];
        temp &= 0xFFFFFFFF;
        E = D;
        D = C;
        //循环左移 30 位
        C = SHA1CircularShift(30,B);
        B = A;
        A = temp;
    }
//第四轮操作,操作过程见图 4-12
    for(t = 60; t < 80; t++)
    {
        //循环左移 5 位操作
        temp = SHA1CircularShift(5,A) + (B ^ C ^ D) + E + W[t] + K[3];
        temp &= 0xFFFFFFFF;
        E = D;
        D = C;
        //循环左移 30 位
        C = SHA1CircularShift(30,B);
        B = A;
        A = temp;
    }
```

```
    //得出结果
    Message_Digest[0] =   (Message_Digest[0] + A) & 0xFFFFFFFF;
    Message_Digest[1] =   (Message_Digest[1] + B) & 0xFFFFFFFF;
    Message_Digest[2] =   (Message_Digest[2] + C) & 0xFFFFFFFF;
    Message_Digest[3] =   (Message_Digest[3] + D) & 0xFFFFFFFF;
    Message_Digest[4] =   (Message_Digest[4] + E) & 0xFFFFFFFF;
    Message_Block_Index = 0;
}
```

4.5.3 计算范例

举例说明使用五种 SHAx 算法进行散列值计算。输入原文为：

FC 1C C7 D6 E8 A4 87 CA AB BE 52 5C 5C B2 A3 2A

采用不同的 SHAx 算法,计算得出的散列值结果为：

(1) SHA1 计算的结果(20 字节)

21 F2 EB 26 11 D9 31 6B 53 41 1F 8E B2 A5 0E 7D
4C A6 4D BA

(2) SHA224 计算的结果(28 字节)

FE 24 A8 BF DC 17 78 53 B1 24 D7 B0 FA 4E 1D D7
97 F6 04 68 2B BA 20 DC A3 DE 10 E3

(3) SHA256 计算的结果(32 字节)

AD 77 9C CC E2 75 10 39 8B AF 7F 22 20 BA 4F 68
51 C8 9B F7 D3 DB F4 CE 73 50 86 52 CB 42 C0 66

(4) SHA384 计算的结果(48 字节)

26 46 96 B2 4A 7B 48 87 86 5A BF 73 8C F0 90 B4
EC 5B 74 25 6C 9C 9A 39 41 73 54 3C 3A B0 58 63
B4 AC E2 F0 5B 8F 76 F2 3D D9 CB FE 7C 68 F0 52

(5) SHA512 计算的结果(64 字节)

A5 A8 2C DC 30 76 68 11 47 EF 64 5B 15 20 E4 F1
6B 73 5B 8A 37 17 64 CC 1B B0 F3 41 AF FC 7F A2
5F BD 2C D1 52 19 8D AB 73 74 89 B6 B1 AD E8 13
D2 AC 63 C8 73 DD BD E1 F4 1D B6 7D 67 CB 53 7C

4.6 MAC 算法

为了保证读写机具和智能卡之间命令交互过程中数据的完整性和私密性,通常将报文鉴别码(message authentication code,MAC)附加在原始报文后一起发送。MAC 码为原始数据通过对称加密算法进行加密运算后产生的 4 字节或 8 字节的鉴别数据。报文鉴别码(MAC)产生和认证过程参见图 4-13。

MAC 码主要有以下功能：

(1) 接收方可以相信接收到的消息未被修改。这是完整性验证。

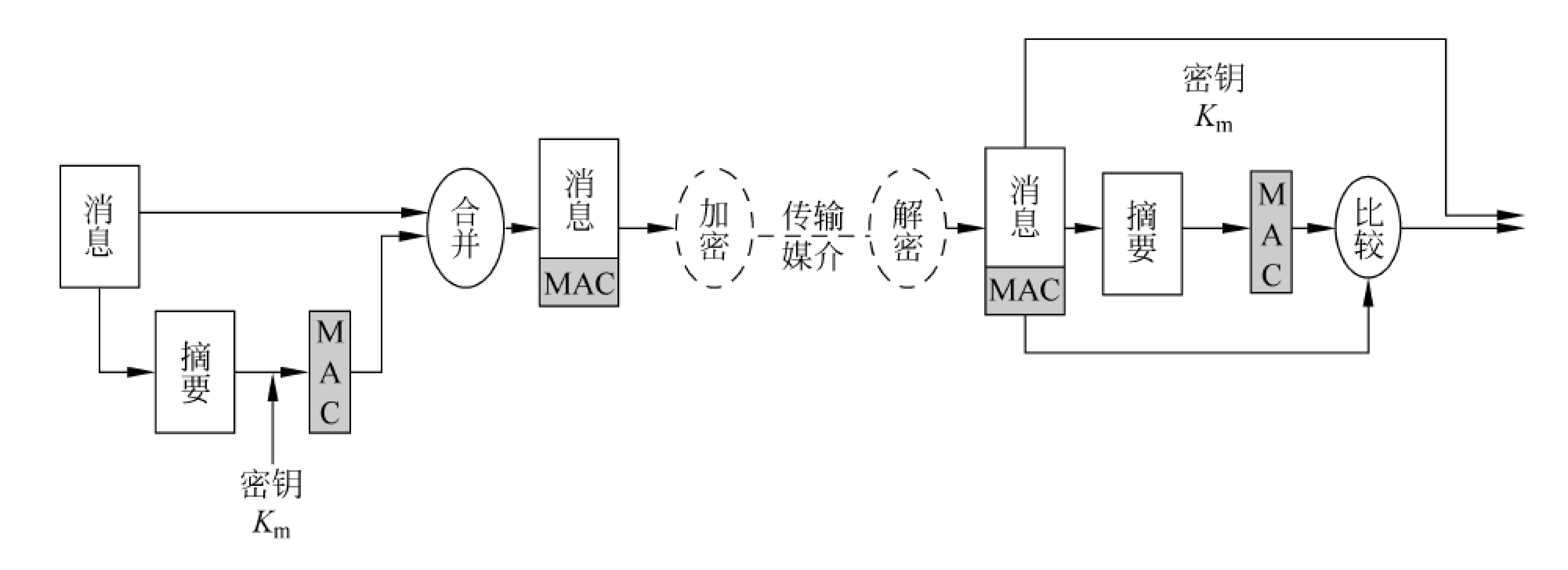


图 4-13 报文鉴别码(MAC)产生和认证过程

(2) 接收方可以相信接收到的消息来自真正的发送方。这是不可否认性验证。

在 ISO/IEC 9797-1 规范中,定义了 6 种 MAC 算法,本节不再赘述。下面列举几种在智能卡应用中常见的 MAC 算法。

4.6.1 FULL DES/3DES MAC

Global Platform 规范的前身是 Visa 组织提出的 Open Platform,是一套专门为实现跨行业智能卡协同工作而提出的规范标准,旨在降低甚至消除不同行业间的多应用智能卡在通信方面的壁垒。

在 Global Platform 规范中定义了安全通道协议(secure channel protocol,SCP),在 SCP01 版本中采用的是 FULL DES MAC 计算方式,流程如图 4-14 所示。

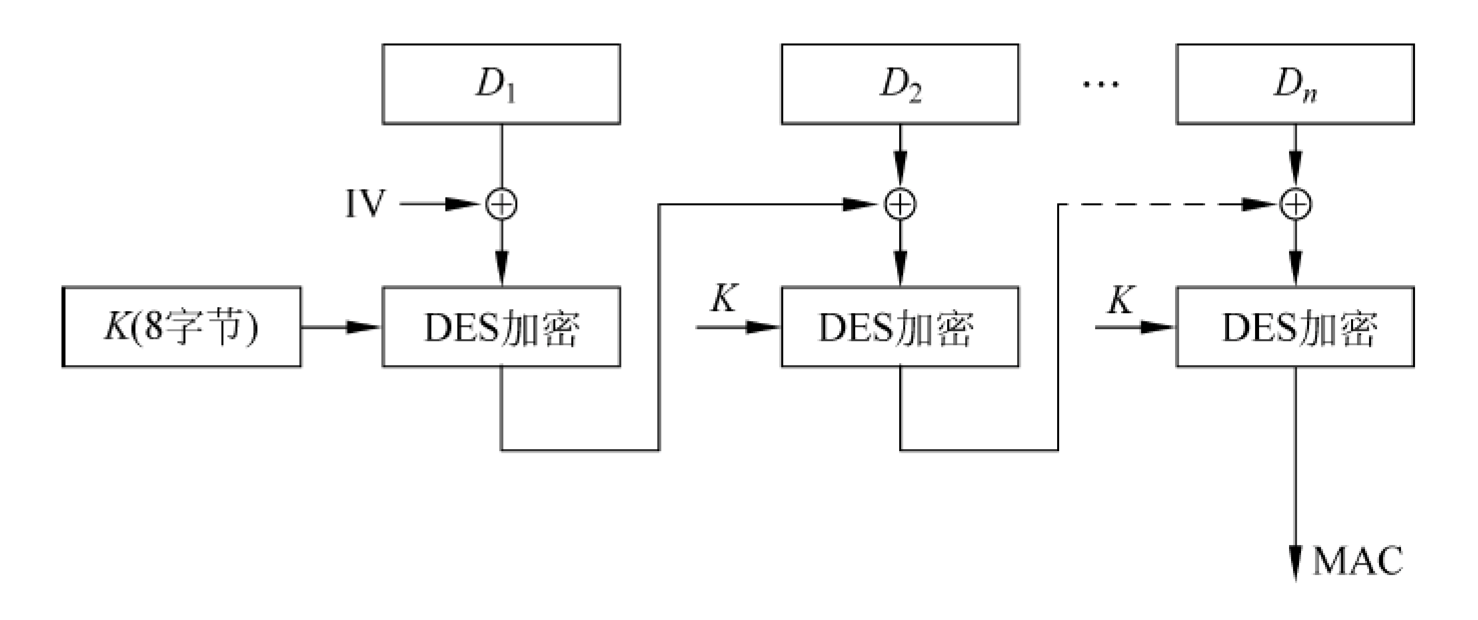


图 4-14 FULL DES MAC 算法结构图

在 SCP02 版本中采用的是 FULL 3DES MAC 计算方式,流程如图 4-15 所示。

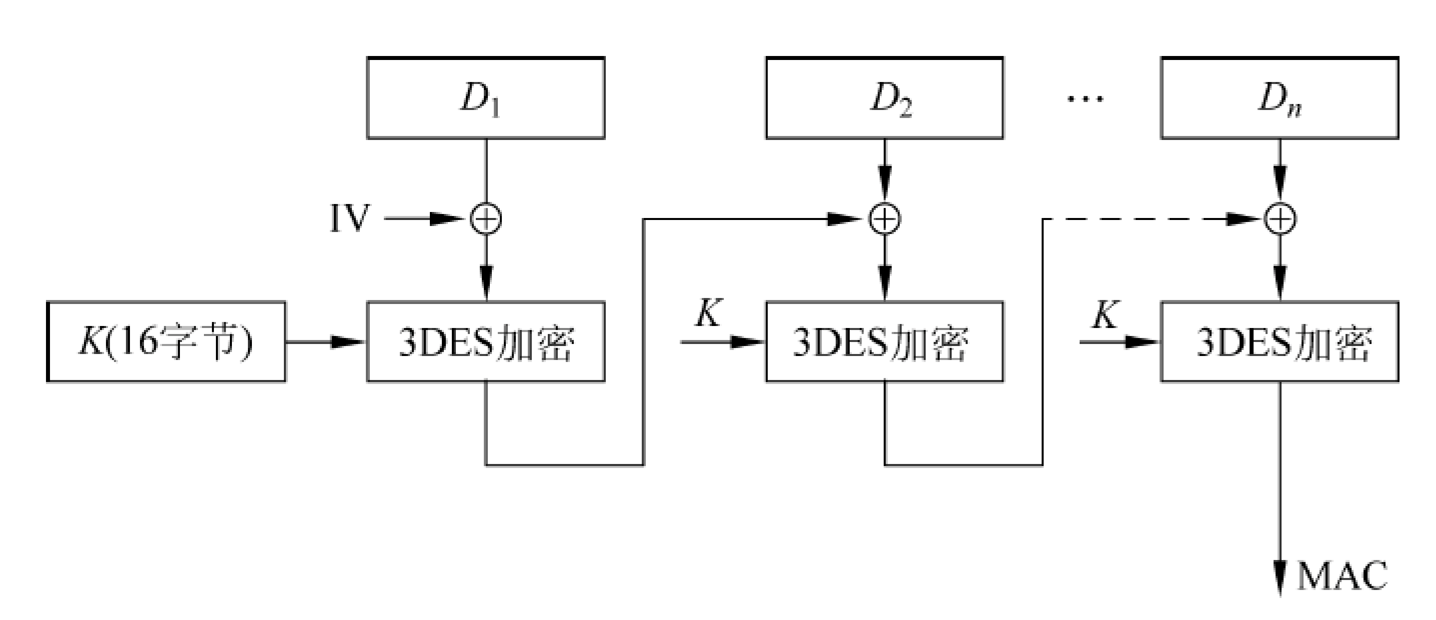


图 4-15 FULL 3DES MAC 算法结构图

4.6.2 Retail 3DES MAC

国际民航组织 ICAO 9303 规范规定电子护照应用中，命令和响应的安全报文 MAC 码采用 Retail 3DES MAC(零售版 3DES MAC)算法，流程如图 4-16 所示。其中发送序列计数器(send sequence counter，SSC)作为数据块 D_1 参与运算。最后结果为 8 字节的 MAC 码。

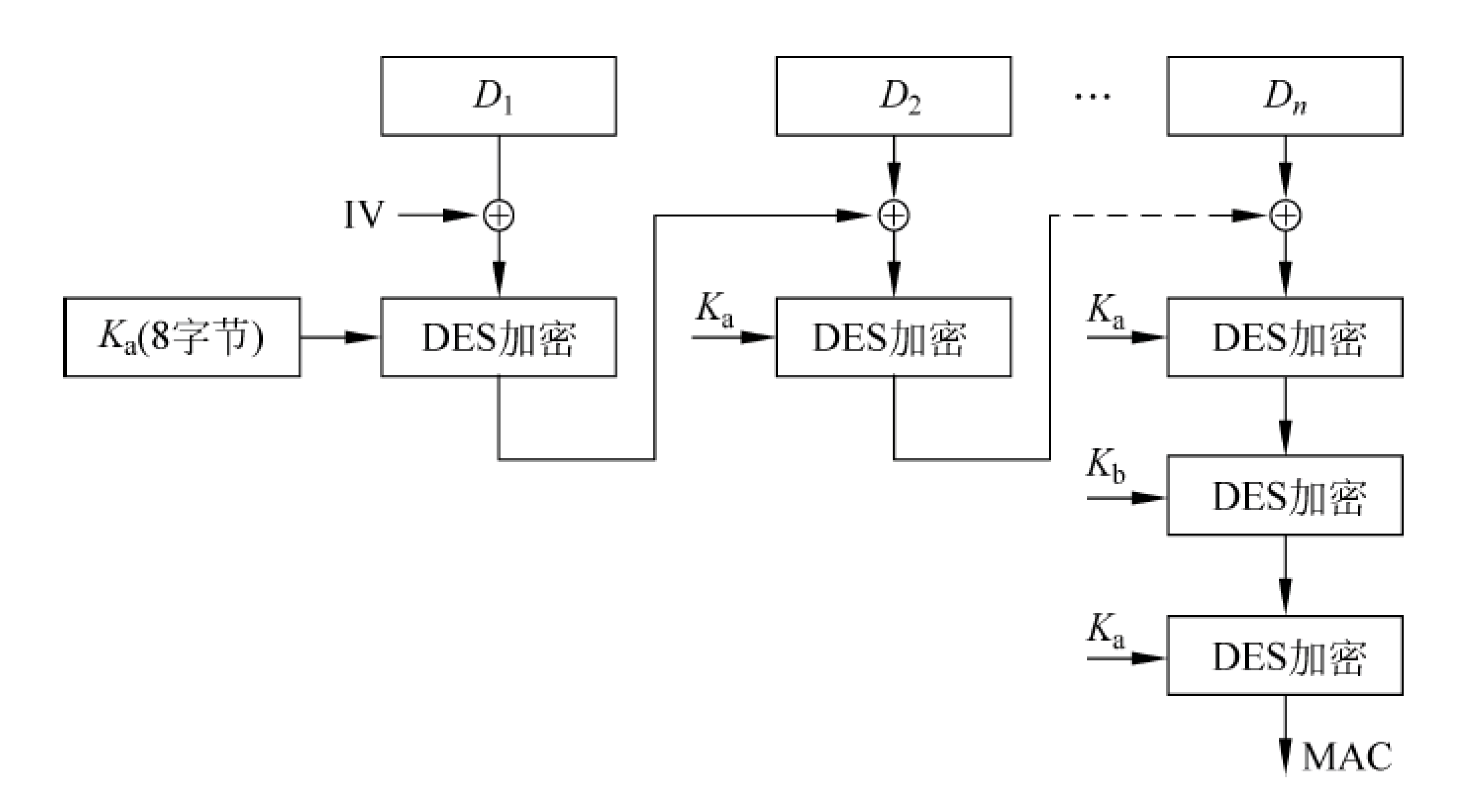


图 4-16 ICAO Retail MAC 3DES MAC 算法结构图

4.6.3 PBOC MAC

中国金融集成电路卡规范 PBOC 2.0 中 MAC 码计算分两种方式：

(1) 基于 64 位分组加密算法的 MAC 计算方法：该方法也采用 Retail MAC 算法，只是最后 MAC 码结果为 4 字节。

(2) 基于 128 位分组加密算法的 MAC 计算方法：采用 CBC 模式的 128 位分组加密算法以及 MAC 过程密钥 K_S 对任意长度的报文 MSG 计算一个 s 字节的 MAC，步骤如下：

- 填充并分块

依据 ISO/IEC 7816-4(等价于 ISO/IEC 9797-1 中的模式 2)对报文 MSG 进行填充，因此在 MSG 的右端强制加上 1 个'80'字节，然后再在右端加上最少的'00'字节，使得结果报文的长度 MSG：=(MSG||'80'||'00'||'00'||…||'00')是 16 字节的整数倍。

随后将 MSG 拆分为 16 字节的块 $X_1, X_2, \cdots, X_K$。

- MAC 过程密钥

MAC 过程密钥 K_s 长度为 16 字节。

- 密文计算

用 MAC 过程密钥以 CBC 模式的分组加密处理 16 字节块 $X_1, X_2, \cdots, X_K$：

$H_i := \mathrm{ALG}(K)[X_i \oplus H_{i-1}]$，这里 $i=1,2,\cdots,K$，其中，H_0 的初始值 $H_0 :=$'00'。

用 $H_{K+1} := H_{KL} \oplus H_{KR}$ 计算 8 字节的块 H_{K+1}，则 MAC 值 S 等于 H_{K+1} 的 s 个最高位字节。

PBOC 2.0 MAC 算法结构如图 4-17 所示。

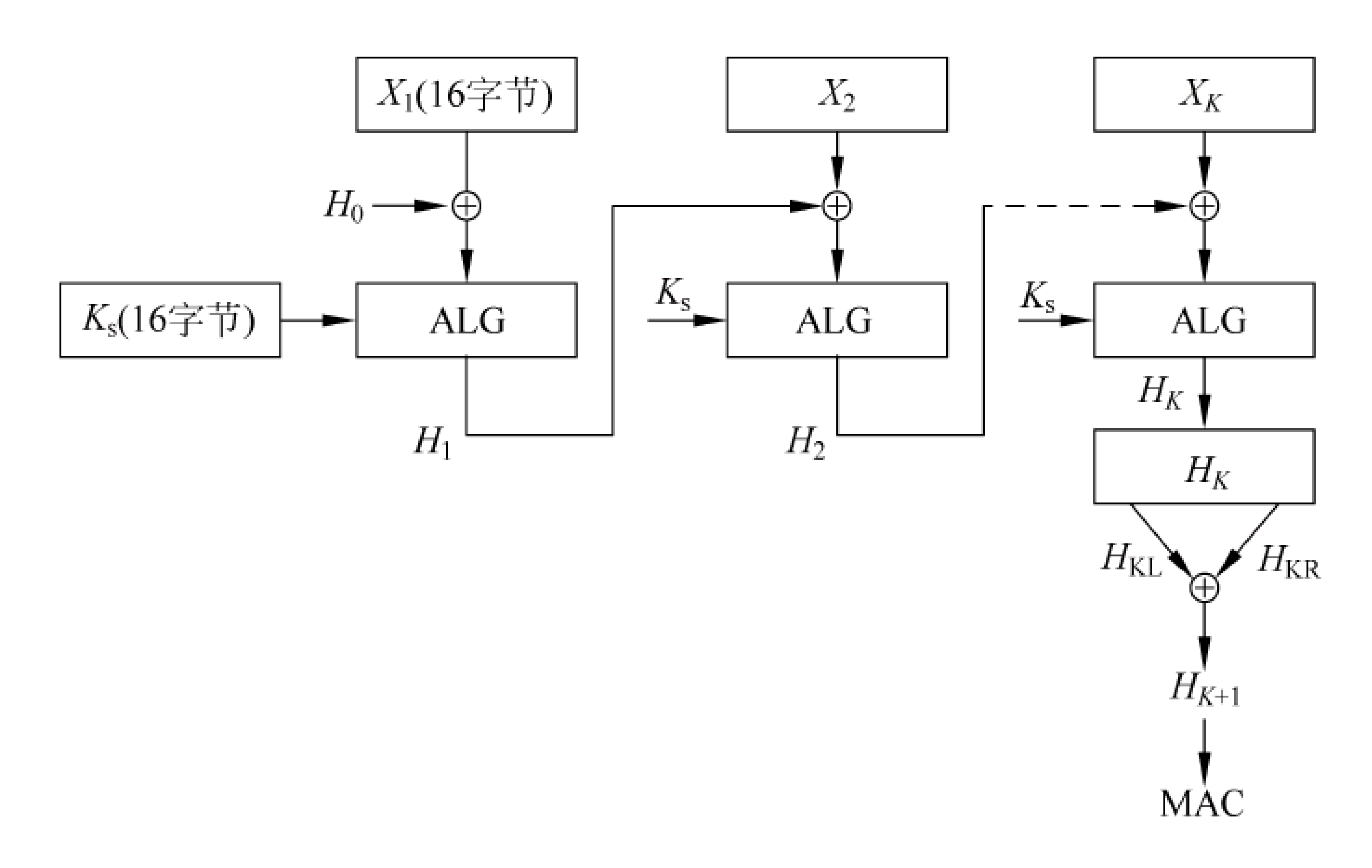


图 4-17 PBOC 2.0 MAC 算法结构图

4.7 国家商用密码算法

国家商用密码算法由国家密码管理委员会制定并管理，算法属于国家机密，资料不公开。在涉及到国家安全等重要领域，智能卡应用需要使用国家商用密码算法，国家商用密码算法的名称、类型和对应的公开算法之间的对应关系如表 4-4 所示。通常国家商用密码算法的实现是通过硬件或软件库实现，软件开发者通常只需调用接口即可。华大、华虹等 IC 芯片制造商均生产含有相关国家商用密码算法的芯片供用户使用。

表 4-4 国家商用密码算法简表

算法名称	算法类型	对应公开算法
SSF33 SSF28	分组密码算法	3DES，AES
SSP02	分组密码算法 信息摘要算法	3DES，AES SHA1
SM1（SCB2）	分组密码算法	3DES，AES
SM2	椭圆曲线密码算法	ECC
SM3（SCH）	密码杂凑算法	SHA256
SM6	分组密码算法	3DES，AES
SM7	分组密码算法	3DES，AES

参考文献

[1] ISO/IEC 9797-1. Information technology. Security techniques. Message Authentication Codes (MACs)，Part 1：Mechanisms using a block cipher，1999

[2] ISO/IEC 9797-2. Information technology. Security techniques. Message Authentication Codes (MACs)，Part 2：Mechanisms using a dedicated hash-function，2002

[3] ISO/IEC 9796-2. Information technology. Security techniques. Digital signature schemes giving

message recovery,Part 2：Integer factorization based mechanisms. Second Edition,2002

[4] ISO/IEC 9796-3. Information technology. Security techniques. Digital signature schemes giving message recovery,Part 3：Discrete logarithm based mechanisms,2000

[5] W. Diffie,M. Hellman. New directions in cryptography. IEEE Transactions on Information Theory,1979,61.22(6)：644～654

[6] R. L. Rivest,A. Shamir,L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystem. Communications of the ACM,1978,21(2)：120～126

[7] 中国金融集成电路(IC)卡借记贷记规范——安全部分. 北京：中国人民银行,2010

[8] Joan Daemon,Vincent Rijmen. The Design of Rijindael AES：The Advanced Encryption Standard. New York：Springer-Verlag Berlin Heidelberg,2002

[9] Sean Murphy. The Advanced Encryption Standard(AES). Information Security Technical Report,1999,4(4)：12～17

[10] Douglas R Stinson 著,冯登国译. 密码学原理与实践. 第 2 版. 北京：电子工业出版社,2003

[11] ICAO DOC9303. Part1 Machine Readable Passports. Volume 2：Specifications for Electronically Enabled Passports with Biometric Identification Capability. Sixth Edition. International Civil Aviation Organization. 2006

[12] Bruce Schneier. 应用密码学. 北京：机械工业出版社,2000

[13] 刘淳,张凤元,张其善. 基于智能卡的 RSA 与 ECC 算法的比较与实现. 计算机工程与应用,2007,(4)：96～98

[14] 周玉洁,冯登国. 公开密钥密码算法及其快速实现. 北京：国防工业出版社,2002

[15] J. Jonsson,Request Comments,B. Kaliski. Public-Key Cryptography Standards (PKCS) ＃1：RSA Cryptography Specification Version 2.1,http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.57.9524 ,2003

[16] Kenji Koyama,Yukio Tsuruoka. Speeding up elliptic cryptosystems by using a signed binary window method. Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology table of contents,1992,740：345～357

[17] A. Bosselaers,R. Govaerts,J. Vandewalle. Comparison of three modular reduction functions. Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology table of contents,1993,773：175～186

[18] 刘淳,张凤元,张其善. 基于智能卡的素数域椭圆曲线密码的快速实现. 计算机工程与应用,2006,(27)：137～139

[19] 袁晓宇,张其善. 两种智能卡芯片的 ECDSA 实现. 计算机工程,2005,31(15)：16～18

[20] 聂景丰. 智能卡操作系统安全模块研究与实现. 成都：西南交通大学,2006

[21] 刘建伟,王育民. 网络安全——技术与实践. 北京：清华大学出版社,2005

[22] Wade Trappe Lawrence C. Washington 著. 邹红霞,许鹏文,李勇奇译. 密码学概论. 北京：人民邮电出版社,2004

[23] 基础性商用密码产品表. 北京市国家商用密码管理办公室,2010



安全机制

智能卡安全包括智能卡的数据安全和数据源宿之间的通信安全。其中,智能卡的数据安全包括数据的安全存储和授权访问。数据传输的安全问题是指:攻击者对智能卡与接口设备通信线路中的信息流进行截听、复制或者修改。安全通信需重点保护通信链路过程中不受到攻击。在典型智能卡应用中,存在4个通信通道,如图5-1中的①、②、③、④通道所示。本书主要讨论其中的第一通道,即智能卡与读写机具之间的安全通信。其他通道的安全涉及到读写机具物理系统的安全、因特网的安全、数据库的安全,均可查阅到相关书籍,本书不再赘述。

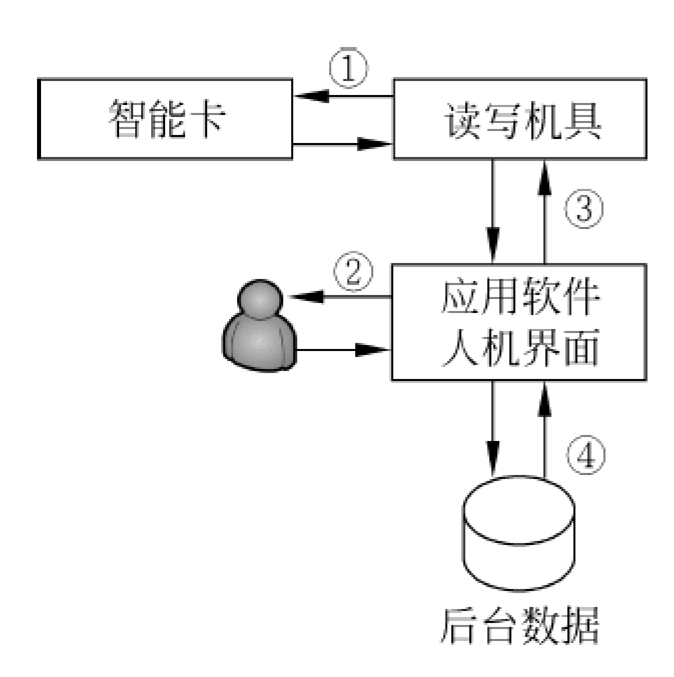


图5-1 典型智能卡应用中的安全通道

5.1 安全机制简介

标准X.800推荐了8种安全机制以便提供第3章描述的6种安全服务:

- 加密:运用安全算法对数据进行变换,使之成为未授权不可知的形式。
- 数据完整性:保证数据元或者数据流的完整性的各种机制,如附加MAC码等。
- 数字签名:附加不可伪造的发送者信息在数据元之后,以保证数据的来源和完整性。
- 身份认证交换:通过信息交换来保证实体身份的各种机制。
- 流量填充:在数据流空隙中插入若干位以阻止流量分析。
- 路由控制:能够为了某些数据动态地或者预定地选取路由,确保只使用物理上安全的链路。
- 公证:利用可信赖的第三方来保证数据交换的某些性质。
- 访问控制:对资源实施访问控制的各种机制。

根据智能卡实际应用的特点,下面分别给出上述安全机制的具体实现细节。

5.2 PKI基础

公开密钥基础设施(public key infrastructure,PKI)是以公钥算法为基础的,包含公开密钥技术、数字证书、权威认证机构、注册机构、证书作废系统、应用接口等组件的提供认证服务的基础设施。PKI为建立在之上的应用提供了机密性、完整性、不可否认性和身份认证的功能。

5.2.1 PKI 结构

根据 PKIX 系列标准中 RFC 2510 的定义，一个完整的 PKI 产品通常应具备以下几个组成部分：最终实体(end entity，EE)、权威认证机构(certification authority，CA)、注册机构(registration authority，RA)、证书库(Repository)。结构如图 5-2 所示。

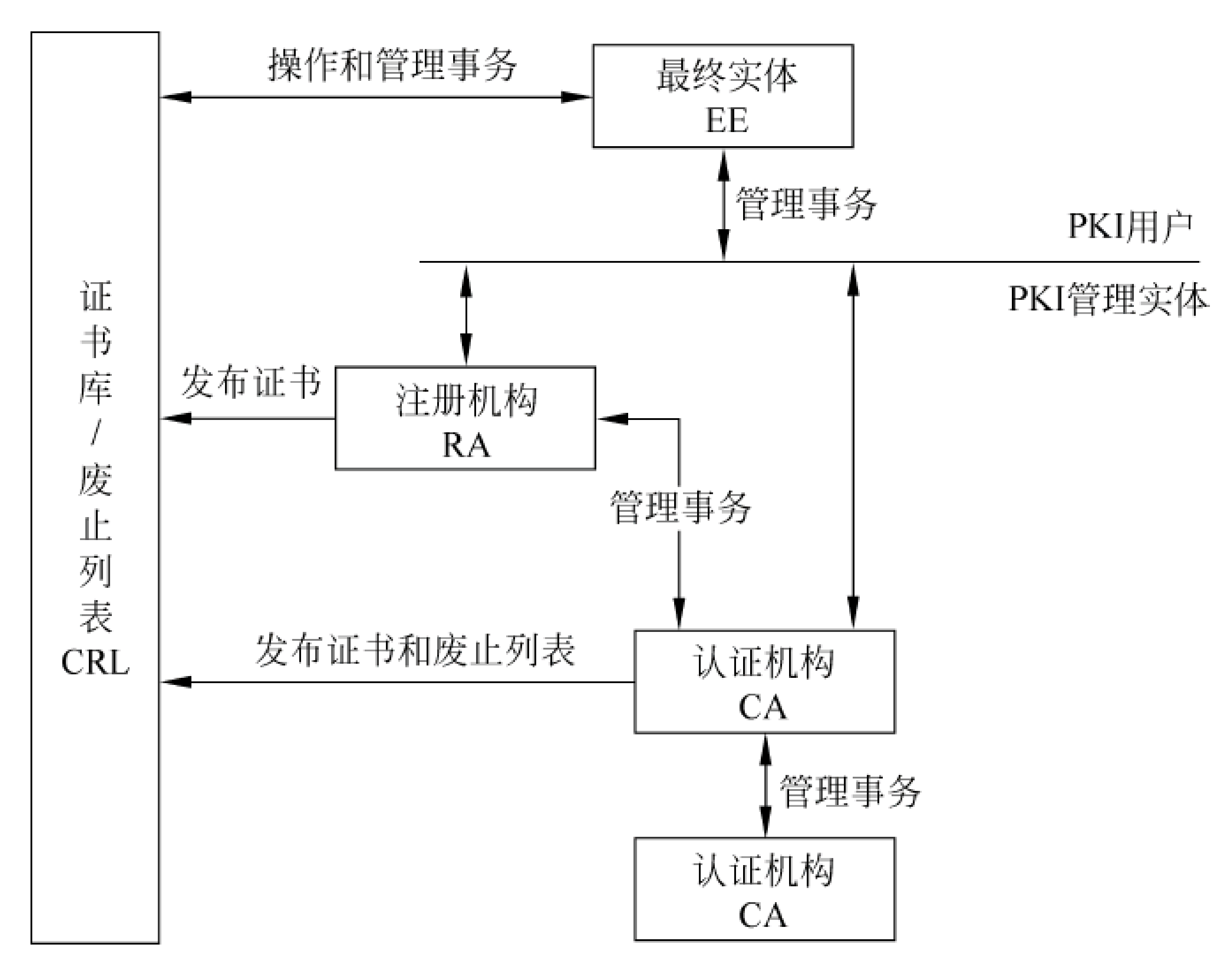


图 5-2　PKIX 标准中的 PKI 体系结构模型

各组件的功能如下：

(1) 最终实体(EE)。PKI 中的用户，可分为两类：证书持有者和依赖方(relying party)。证书持有者即为证书中所标明的用户，依赖方指的是依赖于证书(真实性)的用户。在数字签名中，依赖方是在相信证书真实性的基础上来验证持有者签名的。

(2) 认证机构(CA)。可信权威机构，负责颁发、管理和吊销最终实体的证书。CA 最终负责它所有最终实体身份的真实性。具体功能如下。

颁发数字签名证书：CA 支持三种有关数字签名证书的证书请求，即 RA 发起的注册请求、更新和自我注册请求。根据每种请求的不同，CA 以不同的方式来鉴别这些申请证书主体的身份。

撤销证书：CA 产生和发布证书撤销列表(certificate revocation list，CRL)。CRL 中包括所有被撤销但没有到期的证书。CA 也可以颁发间接的和增量 CRL。被颁发的 CRL 的形式由 CA 的认证业务说明决定。

请求 CA 证书：CA 应能向层次更高的 CA 申请证书，以支持 PKI 的层次信任模型。

(3) 注册机构(RA)。负责在 CA 为实体发放证书以前验证实体身份并在 PKI 系统中为用户注册。RA 是一个可选的组件，假如没有 RA，CA 则必须承担 RA 的功能。主要功能如下。

接受证书申请：RA 系统支持在线和离线两种方式接受来自各证件签发安全服务器提交的证书申请，并对申请实施管理。

证书申请的审核和证书签发：管理人员依据相关政策对证书申请实施审核，只有通过

审核的请求才能提交到 CA 系统签发证书。

撤销证书：当证书需要撤销时，管理人员通过 RA 系统提交证书撤销请求，签发证书撤销列表。

系统数据的备份与恢复：系统数据的备份与恢复在 RA 系统中具有重要意义。当系统受到损坏时，利用已备份的数据可以使系统恢复原状，重新回到可以正常运转的状态。

管理员身份管理、安全审计及统计报表：管理员登录系统需使用数字证书，只有使用该管理员证书才能登录 RA 系统管理服务。管理员登录系统和对该系统所有的操作都被记录在系统日志中，用于安全审计。

(4) 证书库(Repository)。开放的电子站点，负责向所有的最终用户公开数字证书和证书废止列表(CRL)。CA 中心使用的证书都是基于标准证书格式 X.509，在 ITU-T X.509 或者 ISO/IEC ITU 9594-8 中定义。

除上述基本组件之外，一个完备的 PKI 系统，还应包括以下内容。

证书运作声明：整个系统都是按照此声明所定义的各种规范来操作的，它是 PKI 系统的基础；

密钥备份及恢复系统：负责从备份数据加密的密钥历史库中提出备份密钥，以便用户解开加密的历史数据，避免数据的丢失；

证书撤销处理系统：主要负责处理在某种情况下(比如证书主体属性的改变、私钥的泄漏或丢失等)需要对证书进行撤销的操作；

PKI 应用接口：主要用来为用户提供良好的应用程序接口。

5.2.2 数字证书

PKI 中的数字证书相当于网上的身份证，它帮助各个实体有效地识别对方的身份和表明自己的身份。数字证书在一个身份和该身份的持有者所拥有的公/私钥对之间建立了一种联系，它具有以下特点：

(1) 数字证书是 PKI 体系的核心元素。PKI 的核心执行机构是 CA 认证中心，认证中心所签发的数字证书不仅是 PKI 的核心组织部分，而且是 PKI 最基本的活动工具，是 PKI 的应用主体。它完成 PKI 所提供的全部安全服务功能，可以说 PKI 体系中的一切活动都是围绕数字证书进行的。

(2) 数字证书是权威的电子文档。数字证书实际上是由具有可信、公正的第三方权威认证机构签发的。数字证书的内容必须包含权威认证机构的数字签名，即将数字证书的内容做散列杂凑运算后，再用该 CA 机构的私钥，对证书的杂凑值做非对称加密运算，即 CA 对证书的数字签名。CA 对其签发的数字证书内容的签名，是具有法律效力的，是符合国家电子签名法要求，所以数字证书在相互认证过程中是一个公认权威的电子文档。

(3) 数字证书是网上身份的证明。因特网上的身份认证，靠证书机制实现身份的识别与鉴别，因为数字证书的主要内容就包括证书持有者的真实姓名、身份唯一标识和该实体的公钥信息。电子认证机构 CA 是通过对实体签发的这个数字证书，来证实该实体在网上的真实身份的。

(4) 数字证书是 PKI 体系公钥的载体。公钥基础设施是靠公/私钥对的加/解密运算机制完成 PKI 服务的。私钥严格保密，公钥方便公布。方便地传递和发布公钥是公钥基础设

施的优势。公钥发布或传递的方式：一是靠 LDAP 目录服务器，即将 CA 签发的证书发布在目录服务器上，供需进行通信的证书依赖方索取；二是可由通信双方的一方，将公钥证书与加密(签名)后的数据一起发送给依赖方的证书用户。这种公钥的传递载体就是数字证书。

最广泛接受的证书格式是由 ITU-T(国际电信联盟电信标准化组织)定义的 X.509 国际标准定义的。X.509 v3 证书格式在 PKI 中得到了广泛应用，其内容包括证书序列号、证书持有者名称、证书颁发者名称、证书有效期、公钥、证书颁发者的数字签名等。

PKI 数字证书可提供 4 种重要的安全保证：

(1) 机密性(confidentiality)

PKI 所提供的机密性是指发送方将信息经过接收方证书的公钥加密后的密文信息，只能由接收方用自己的私钥解密。在加密应用中，任何想给接收者发送保密信息的人都可以公开、自由地得到接收者的公钥；而接收者的私钥则是需要严格保护的，绝对不能泄漏给其他人。

(2) 完整性(integrity)

数据的完整性是指信息的接收者能够检验收到的信息是否被篡改。由于 Hash 运算在证书应用中是一个必须的手段，用户验证签名后的 Hash 值可以很容易判断是否被篡改。

(3) 不可否认性(no repudiation)

为了防止人们否认自己曾经做过的事情，PKI 为此做了安全而有效的防范措施。因为只有你能使用自己的私有密钥，所以当你对所做的事情留下签名时，你就无法否认所做的事情。

(4) 鉴别(authentication)

身份认证是数字签名技术的典型应用，由于证书机构的权威性，其颁发的公钥证书实际上代表了证书持有人的真实身份。从某种意义上说，公钥证书是证书持有者的数字化身份证，而证书持有人的私有密钥与证书的公钥是一一对应的，因此用私有密钥对数据进行的加密(即数字签名)，毫无疑问就代表了证书持有人对数据进行的加密。任何人收到加密数据以后，如果能够用某人的公钥证书解密数据，则表示该文件是由此证书的持有人所发送的。

5.2.3 数字签名

数字签名是指使用密码算法，对待发的数据进行加密处理，生成一段杂凑值信息附加在原文一起发送，这段信息类似于现实中的签名或印章。它提供了一种鉴别方法，以解决伪造、抵赖、冒充等问题。数字签名在信息安全中，包括身份认证、数据完整性、不可否认性等方面的重要应用，特别是在大型网络安全通信中的密钥分配、认证以及电子商务系统中具有重要作用。数字签名必须保证以下 3 点：

(1) 接收者能够确认发送者对报文的签名。

(2) 发送者事后不能抵赖对报文的签名。

(3) 任何人都不能伪造对报文的签名。

数字签名共涉及三方实体：签名者、验证者和认证服务中心，业务流程如图 5-3 所示。签名者使用签名模块在终端环境中产生签名；验证者使用验证模块在终端环境中验证签名结果，以确认签名者的身份并确定签名者对被签名数据电文的认可；认证服务中心作为可信的第三方机构，向签名者和验证者提供真实性和可靠性服务。除此之外，认证服务中心提供者还向签名者和验证者提供诸如可信时间戳服务、安全资料库服务、公证服务、归档服务等一系列服务。

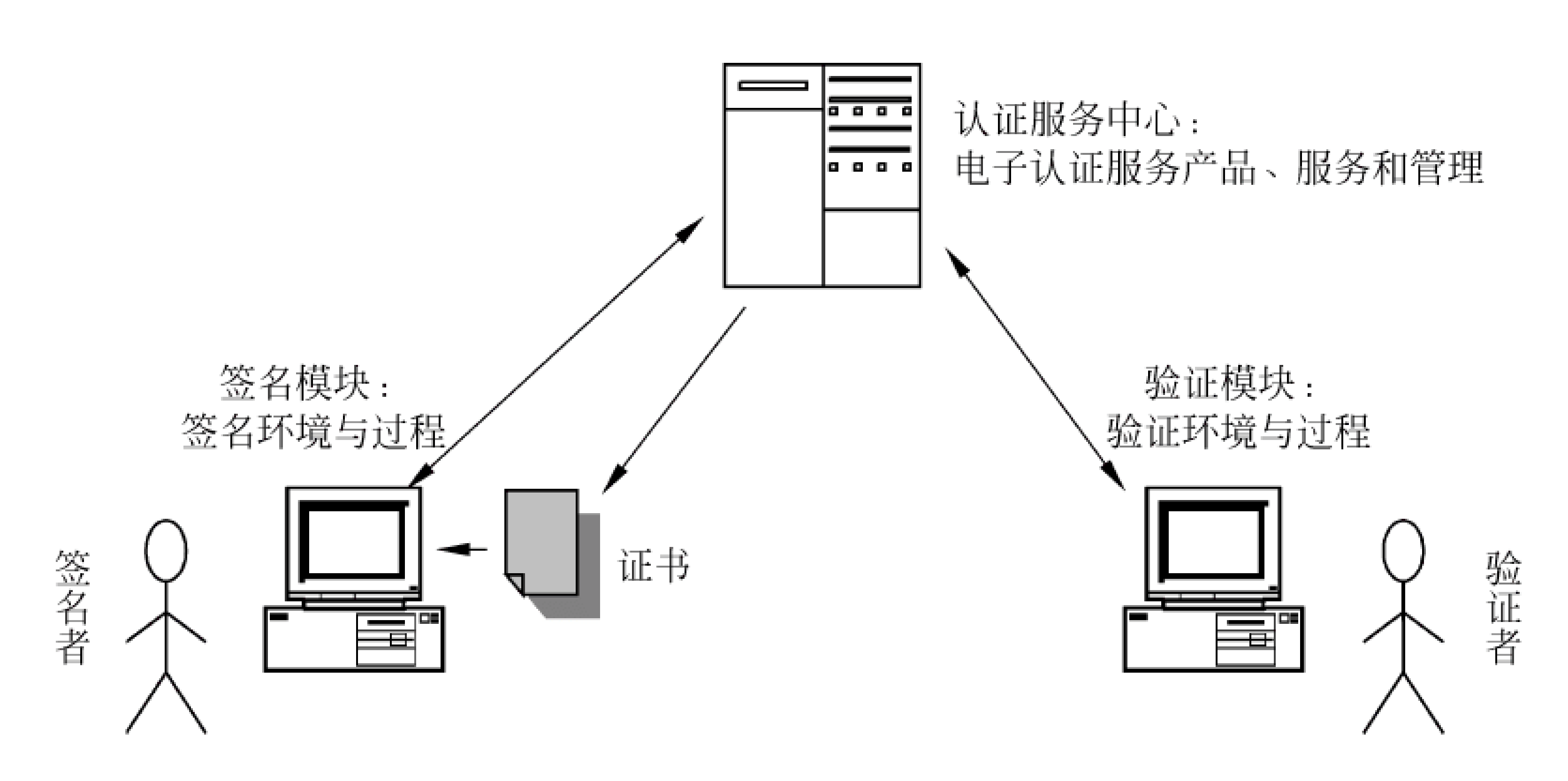


图 5-3 数字签名实体模型

5.3 安全存储

智能卡作为数据存储的载体，存在两方面的安全问题：

(1) 智能卡内的数据被篡改，例如利用逆向工程法分析智能卡内的内容。

(2) 智能卡被伪造。伪造智能卡对攻击的技术要求比较高，需要攻击者具备智能卡制造的相关知识和一定的技术。

智能卡数据安全包括数据的安全存储和安全访问。其中安全存储涉及物理层和软件逻辑层两个层面的保护。

5.3.1 硬件层防护

防范对智能卡芯片的物理攻击，是整个智能卡应用系统安全性的基础。据某些国际知名智能卡芯片厂商称他们的芯片内部具有 50 种以上的安全防护措施。

目前，在芯片制造过程中，防范对 IC 芯片攻击的几种技术为：

(1) 反向工程防范。将芯片的诸多功能随机地分布在微处理器的各个逻辑层中，同时将易于分析的部分(特别是 ROM)尽可能地隐藏于较低层。在芯片中嵌入伪部件和伪功能。内存部分存储时加密。

(2) 总线加密和不规则编址。搅乱总线顺序，且不同层有不同顺序。存储器排列顺序与逻辑地址顺序不同。

(3) 传感器网络。IC 芯片内部设计有电压/频率/光照/温度/干扰传感器。相应的检测功能。当芯片内置检测器监测到芯片工作参数低于/高于设定阈值时，检测器将控制芯片停止工作。

5.3.2 软件层防护

COS 层对数据进行软件层的保护，主要有以下措施：

(1) COS 可将数据加密后存储在 EEPROM 中。

(2) 划分 EEPROM 为秘密区、安全区、透明区，对逻辑地址进行判断和访问。

(3) 使用严格的鉴别和认证,判断外部设备是否有授权。

(4) 对命令的使用条件、生命周期进行严格限制。每条 APDU 命令均对应相应的生命周期,限制其执行范围。智能卡生命周期的定义可参见标准 ISO/IEC 7816-8,各条指令与生命周期的关系如图 5-4 所示。

(5) 掉电保护,防插拔处理。

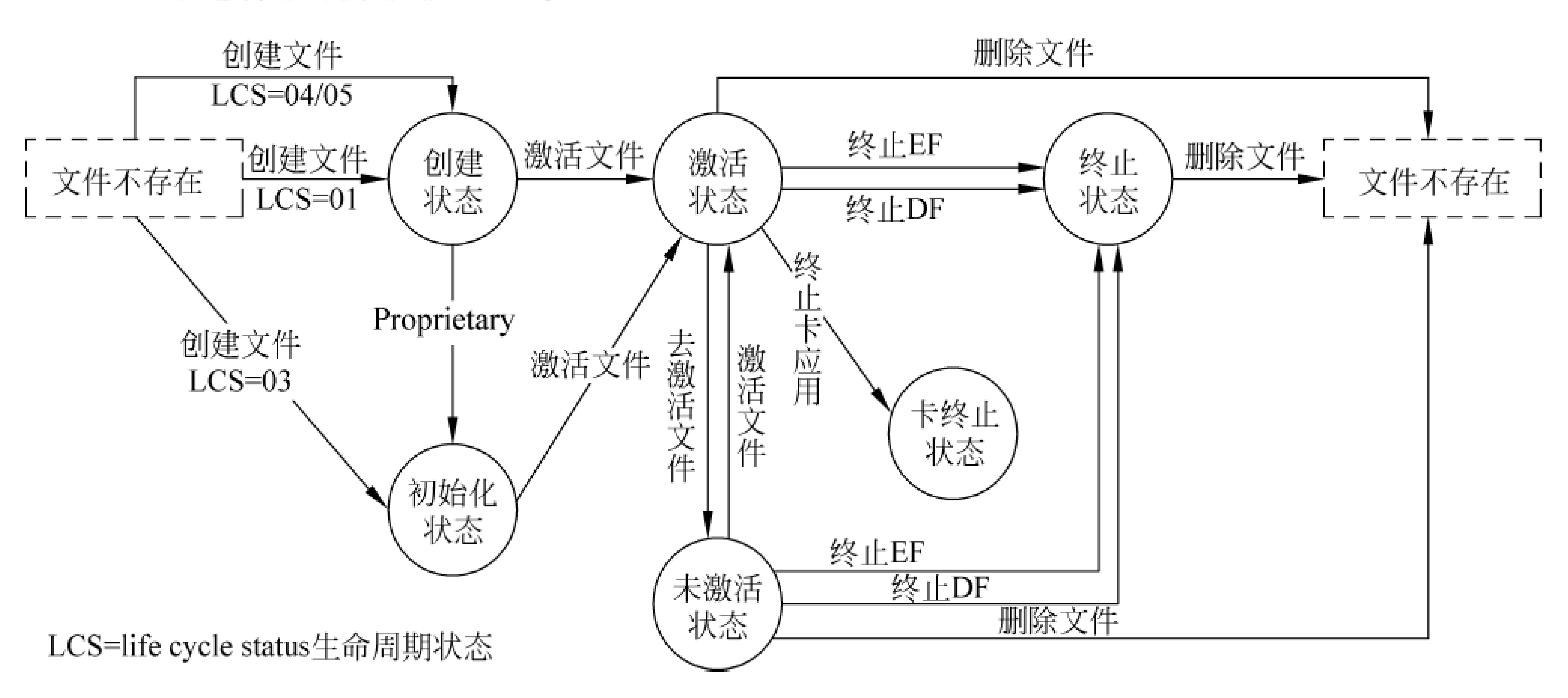


图 5-4 智能卡生命周期

5.3.3 防掉电处理

防掉电操作是为了保证数据完整性的一种方法。防掉电区中保存的是将要被改写的数据。在对 EEPROM 进行写操作之前,先将要更改的数据写入防掉电区,然后将新数据写入到指定位置,操作成功后再将防掉电区清除。

防掉电操作流程如图 5-5 所示。

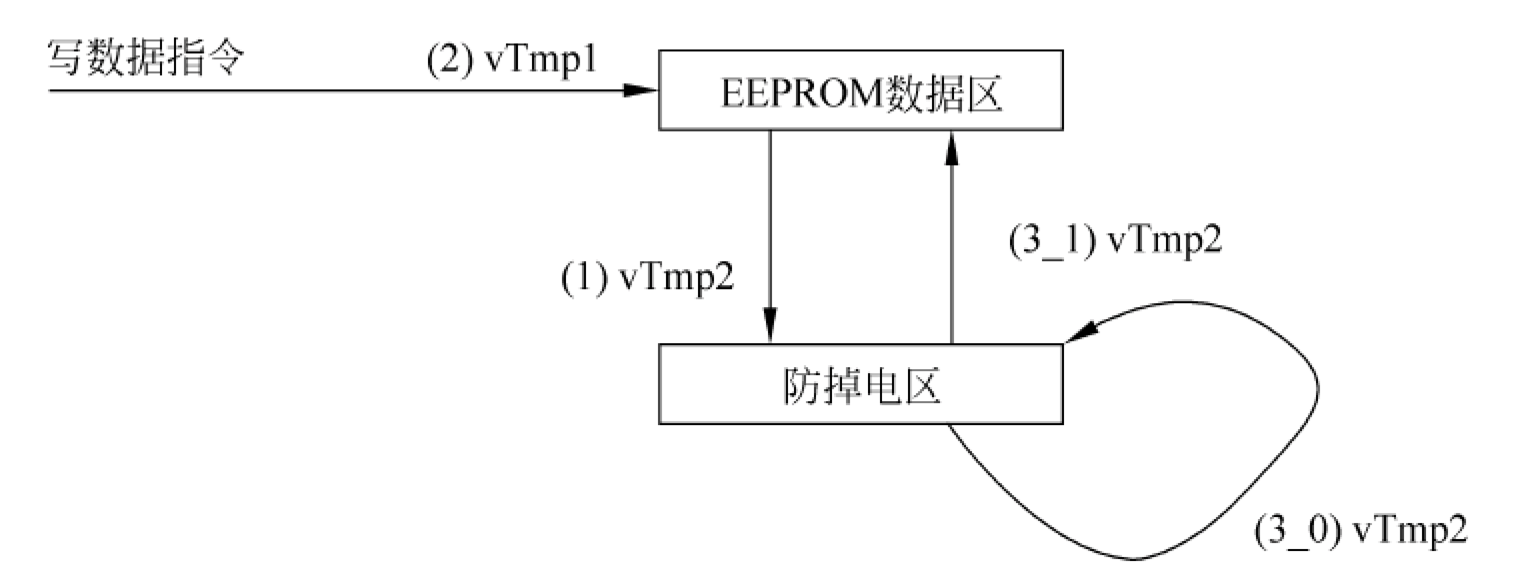


图 5-5 防掉电流程

具体描述如下:

第一步:写数据指令到来时,首先把 EEPROM 中的原有数据备份到防掉电区。见图 5-5 中(1)。

第二步:将新数据写入到 EEPROM。见图 5-5 中(2)。

第三步:

如果第二步正确,则把防掉电区的数据擦除。见图 5-5 中(3_0)。

如果第二步错误,则把防掉电区中的数据回写到 EEPROM。见图 5-5 中(3_1)。

5.4 认证

认证是智能卡和外部系统之间进行身份验证的最重要的一种方式，双方之间的认证最终是通过对被认证方是否正确拥有某一个密钥或者其他私有特征的验证来完成的。

认证按认证对象的不同，主要有以下认证方法：

(1) 对持卡人已知的秘密或者特征(如口令、个人识别号(personal identification number，PIN)、密钥和生物特征)进行的认证；其中生物特征在本书第6章详细描述。

(2) 对读写机具和智能卡的认证，验证机具和智能卡是否拥有某密钥。认证包括外部认证、内部认证和相互认证。外部认证是智能卡对外部系统的认证，外部认证的结果影响卡片相应的安全状态。内部认证是外部系统对智能卡的认证，内部认证结果不影响卡片相应的安全状态。相互认证则是智能卡和外部系统之间相互认证的过程。

5.4.1 PIN 验证

个人识别码(PIN)通常以口令的方式存在。用户将PIN输入到终端，终端将PIN发送到智能卡中，智能卡把输入的PIN和卡内存储的PIN进行比较。如果相同，则验证过程通过，卡片改变内部安全状态。如果不同，则验证过程失败，卡片将重试计数器递减。如果重试计数器减至为0，则卡片锁住，无法再次使用。PIN验证流程如图5-6所示。

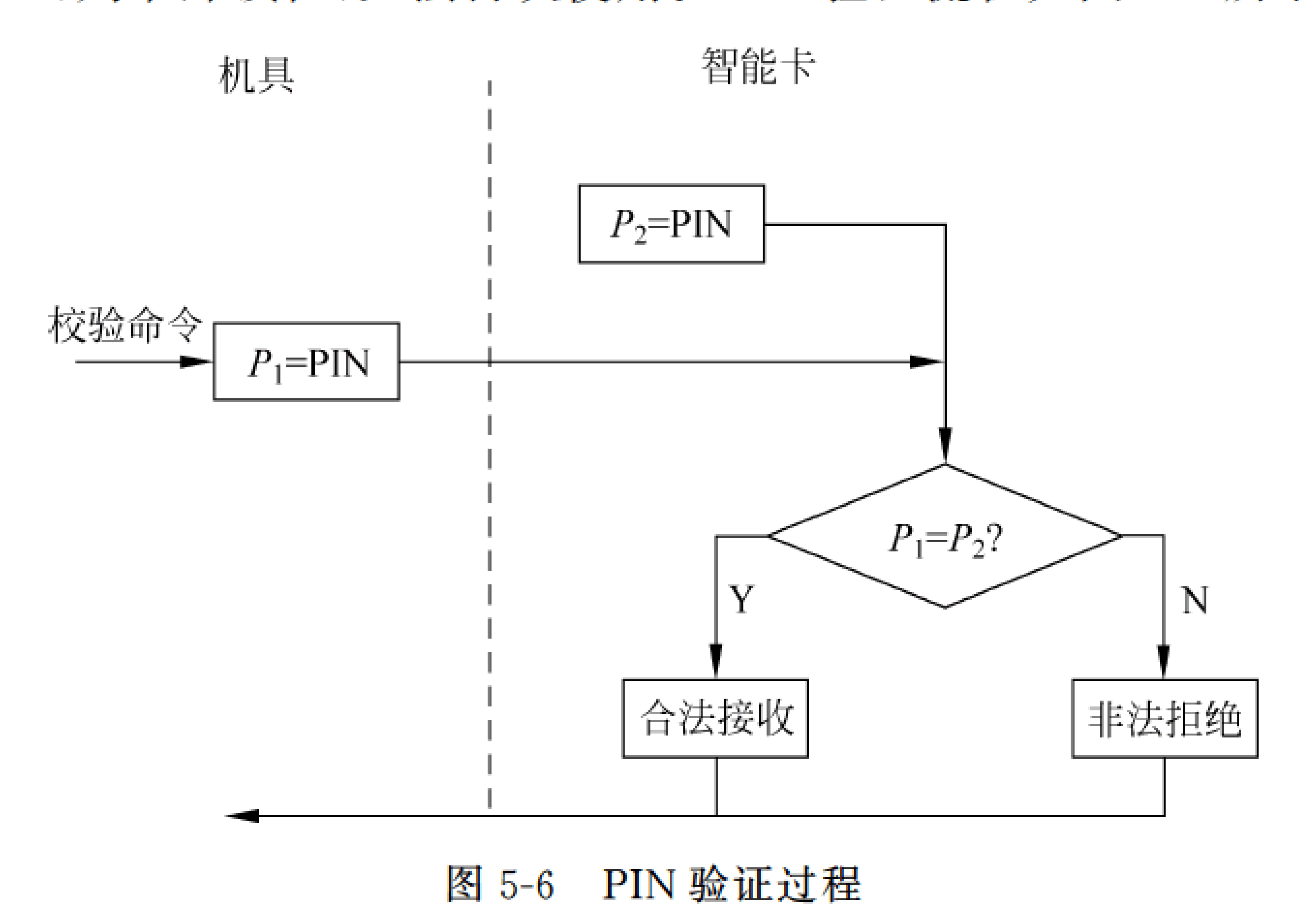


图5-6 PIN验证过程

常见的银行卡ATM取款机多用此方式。如果PIN验证失败到三次，则ATM机吞入该卡，防止卡片PIN码被恶意攻击。

5.4.2 外部认证

外部认证用于智能卡对外部接口设备的认证，检测读写机具是否是合法的外部设备。外部认证有两种认证方式。

(1) 基于对称密钥的外部认证模式

首先，读写机具向智能卡发送取随机数指令，智能卡产生随机数；然后，由读写机具用密钥对随机数加密，以命令的形式将密文发送给智能卡(具体由外部认证命令来完成)；最后，智能卡执行该命令时，将密文解密(用的密钥与读卡器加密用的密钥相同)，并将解密后

的明文与原随机数比较。若两者一致，则证明读写机具是合法的，否则读写机具是非法的。

在 EMV 规范、PBOC 2.0 规范和电子护照 DOC 9303 规范中，均采用对称密钥认证方式，其外部认证流程如图 5-7 所示。

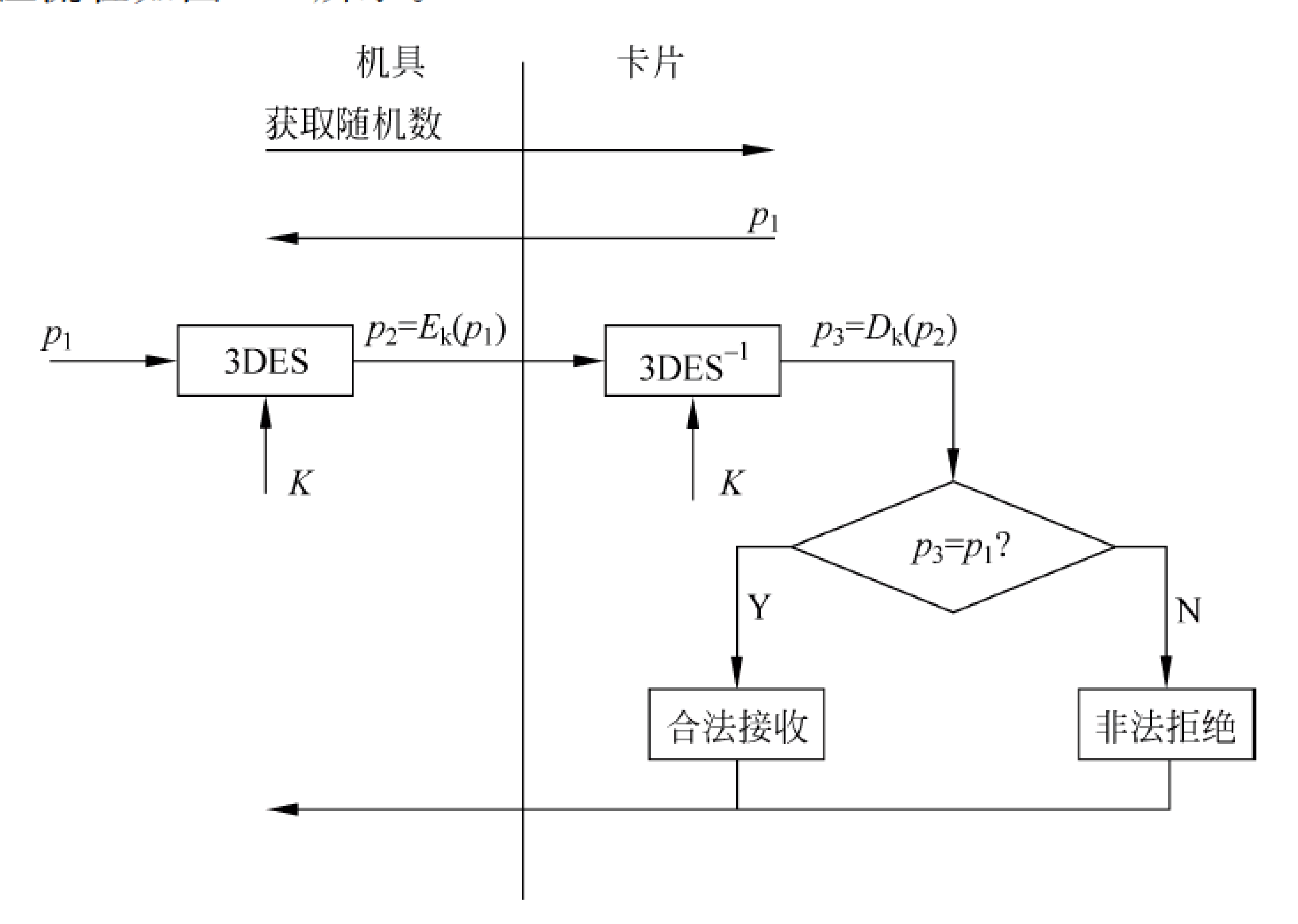


图 5-7 基于对称密钥的外部认证流程图

(2) 基于非对称密钥的外部认证模式

首先，读卡器向智能卡发送取随机数指令，智能卡产生随机数；然后，由读卡器用密钥对随机数签名，以命令的形式将签名发送给智能卡（具体由外部认证命令来完成）；最后，智能卡执行该命令时，认证签名，并将得到的明文与原随机数比较。若两者一致，则证明读卡器是合法的，否则读卡器是非法的。根据签名认证原理，公钥可以认证出签名的真伪，伪造的读卡器是不可能拥有合法读卡器的私钥的，由此可以证明读卡器的真实性。

两种认证模式都可以验证读卡器是否合法，但是它们各自有自己的优缺点。对称密钥密钥长度短，在卡片中存储方便，但安全性差。非对称密钥密钥长，并且不同的机具具有不同的公私密钥对，而卡片 EEPROM 容量有限，无法存储过多的公钥。所以在通常情况下，外部认证使用对称密钥方式。

5.4.3 内部认证

对智能卡进行真伪性认证的过程是内部认证的过程，它利用智能卡唯一的密钥进行验证，伪造的卡片无法具有相同的密钥。

目前内部认证按照密钥类型可分成两种方式：

(1) 基于对称密钥的内部认证模式

首先，读卡器产生随机数；接着，读卡器以命令的形式将随机数发送给智能卡（具体由内部认证命令来完成，与图 5-7 的外部认证方式中的角色正好相反）；然后由智能卡用密钥对随机数加密，并将加密后的随机数发给读卡器；最后，读卡器收到密文，将密文解密（用的密钥与读卡器加密用的密钥相同），并将解密后的明文与原随机数比较。根据对称加密的原理，读卡器的密钥肯定也与智能卡内的密钥相等，伪造的智能卡是没法取得正确的密钥的，由此可以证明智能卡的真实性。

(2) 基于非对称密钥的内部认证模式

首先,读卡器产生随机数;接着,读卡器以命令的形式将随机数发送给智能卡(具体由内部认证命令来完成);然后由智能卡用私钥对随机数签名,并将签名结果发给读卡器;最后,读卡器收到签名,用公钥认证签名。采用非对称密钥,那么密钥是不相同的。公钥是外界知道的,但私钥只有智能卡知道,根据签名认证原理,公钥可以认证出签名的真伪,伪造的智能卡是不能拥有合法智能卡的私钥的,由此可以证明智能卡的真实性。

分析和比较两种认证模式,它们各自有自己的优缺点,如表5-1所示。

表5-1 两种内部认证模式比较

认证方式	基于对称密钥的内部认证	基于非对称密钥的内部认证
密钥	对称密钥	非对称密钥
优点	运算快,执行时间短	运算慢,执行时间长
缺点	密钥管理困难	密钥管理简单
安全性	低	高
可行性	对称密钥难于分发管理,安全性低,所以可行性差	非对称密钥计算比较慢,但是由于认证数据较短,密钥易于分发管理,安全性好,所以可行性好

在电子护照DOC 9303规范中,内部认证采用非对称密钥的方式进行,内部认证命令也称为ICAO Active Authentication,即ICAO主动认证,简称为AA认证。电子护照接收接口设备发来的数据,利用自身存储的相关私有密钥进行签名后返回给接口设备,设备利用公钥进行认证,查看护照是否是合法的护照。

ICAO规定的内部认证流程如图5-8所示。

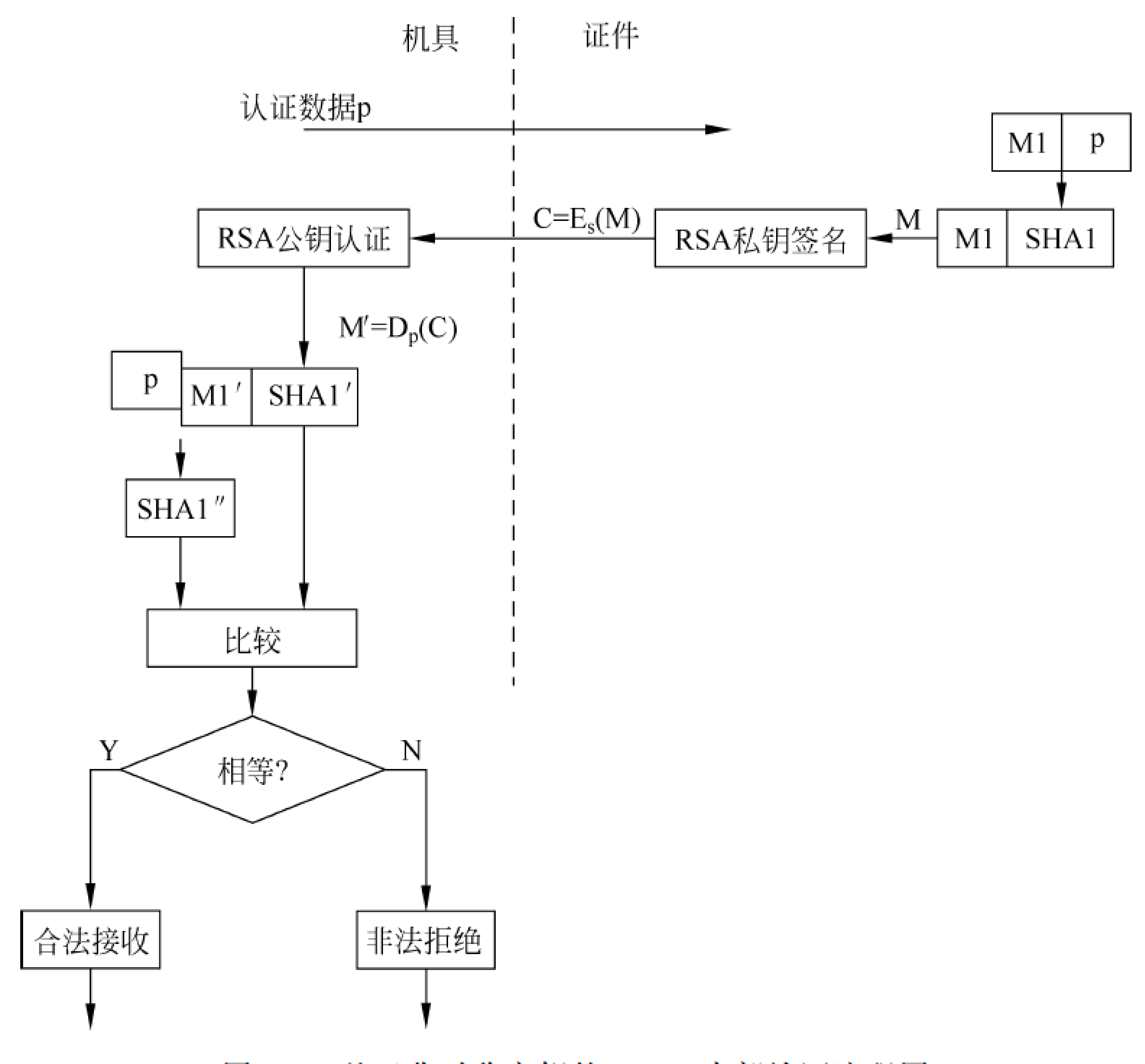


图5-8 基于非对称密钥的ICAO内部认证流程图

5.4.4 相互认证

相互认证过程是智能卡和外部读写机具之间相互认证的过程，可采用外部认证和内部认证结合执行。很多应用中，相互认证除了外部认证和内部认证功能外，还兼有产生会话密钥，进行链路保护的辅助功能。典型范例参见本书 5.5.2 节。

5.5 基本访问控制

电子护照(e-passport)是在普通护照上增加智能卡芯片而成，可存储姓名、性别、生日和出生地等个人信息以及指纹、脸部图像和虹膜等生物特征信息，是加强出入境管理、有效地解决伪造变造、防止偷渡、预防外来恐怖活动的重要技术手段。

电子护照安全主要有以下需求：

(1) 证件资料访问控制、访问授权与认证。

(2) 保护电子数据的完整性、有效性、真实性。

(3) 防止电子数据的恶意复制(克隆)。

(4) 持证人隐私保护(防止非法浏览、非法窃听)。

(5) 其他生物特征资料保护。

电子护照相对于传统护照有两点不同：

(1) 电子护照无需打开，可以直接从证件中读取数据。这种方式被称为掠读(skimming)。

(2) 芯片和读卡器之间的数据没有加密，可以被监听。这种方式被称为窃听(eavesdropping)。

这两种方式都会导致护照中的信息泄漏，存在严重的安全隐患，所以基本访问控制(basic access control，BAC)机制被提出。

该机制的基本思想是：

(1) 将可视化的信息(例如 MRZ 码)分散为控制密钥(加密密钥 K_{ENC} 和 MAC 密钥 K_{MAC})，这样就可以抵御“掠读攻击”。

(2) 利用加密密钥 K_{ENC} 和 MAC 密钥 K_{MAC} 生成会话密钥，每次通信报文都利用会话密钥加密和计算 MAC，形成安全报文格式。这种链路加密方式，可有效抵御“窃听攻击”。

5.5.1 MRZ 密钥分散

护照信息页如图 5-9 所示，信息页中的打印信息包括护照持有人的个人信息(护照号、姓氏、名字、性别、出生地、出生年月日、护照签发日期、签发单位、护照有效期、签发单位)、照片和底部两行机读码(machine readable zone，MRZ)。

两行 MRZ 码的数据结构如表 5-2 所示，每个字段域信息如果字节不够，用“<”填充。

MRZ 码的第二行包括信息字段：证件号码、国家、生日、性别、附加数据。其中除了国家和性别，其他信息字段都计算校验码，然后再利用这四个信息字段和校验码计算一个总校验码。

利用 MRZ 码的第二行中的前三个信息字段和校验码，产生用于 BAC 的密钥种子。具体的流程如图 5-10 所示。



图 5-9　护照信息页

表 5-2　MRZ 信息结构

<table>
<tr><td colspan="12">第一行</td></tr>
<tr><td colspan="3">字段名称</td><td colspan="3">证件类型</td><td colspan="3">发行组织</td><td colspan="3">姓名</td></tr>
<tr><td colspan="3">字节长度</td><td colspan="3">2 字节</td><td colspan="3">3 字节</td><td colspan="3">39 字节</td></tr>
<tr><td colspan="12">第二行</td></tr>
<tr><td>字段名称</td><td>证件号</td><td>校验 1</td><td>国家</td><td>生日</td><td>校验 2</td><td>性别</td><td>期限</td><td>校验 3</td><td>附加数据</td><td>校验 4</td><td>总校验</td></tr>
<tr><td>字节长度</td><td>9</td><td>1</td><td>3</td><td>6</td><td>1</td><td>1</td><td>6</td><td>1</td><td>14</td><td>1</td><td>1</td></tr>
</table>

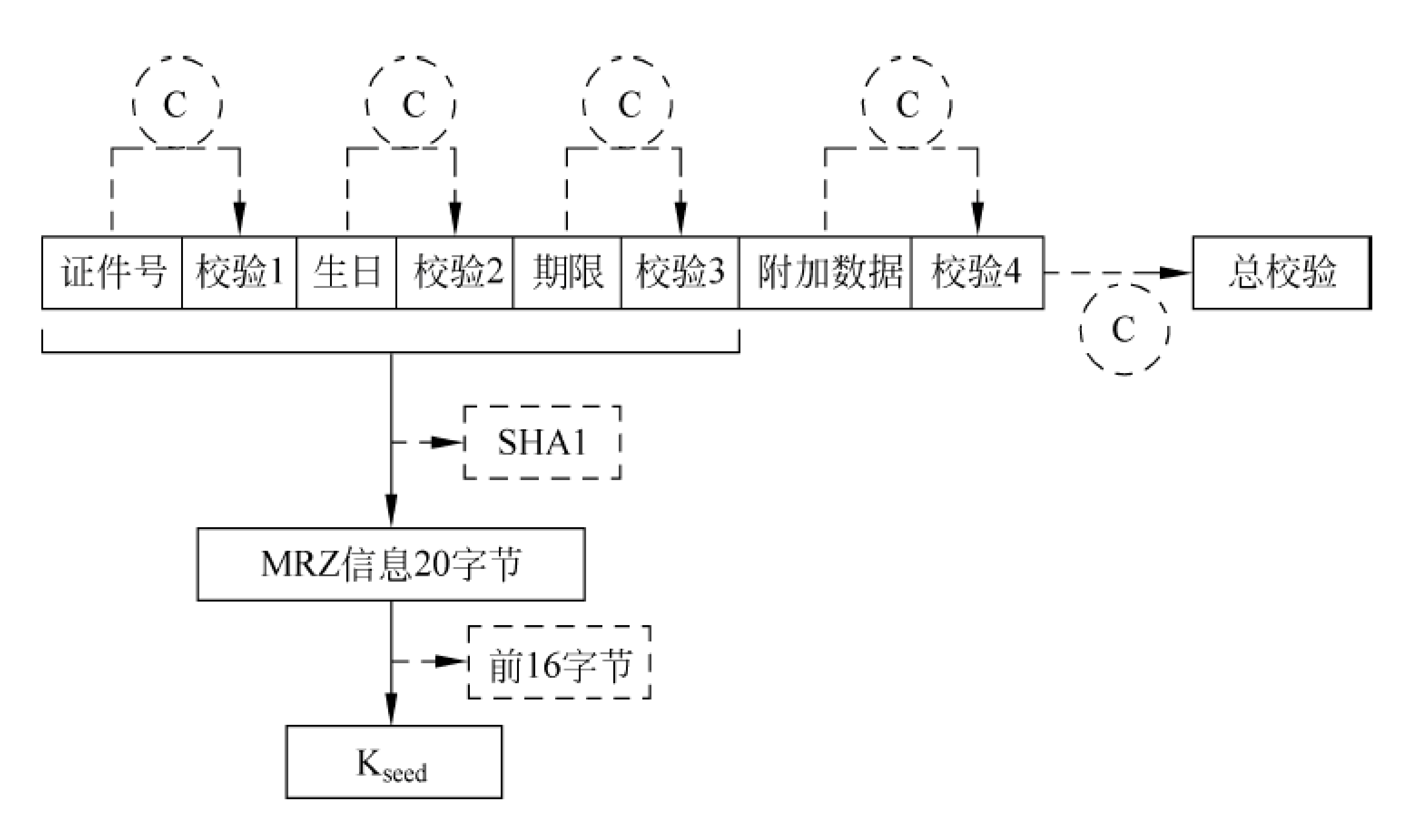


图 5-10　MRZ 校验和密钥种子生成过程

信息域中含有的字符为“0～9”,“A～Z”和“＜”,将它们分别编码,编码规则如表 5-3 所示。

表 5-3 MRZ 信息字符编码

字符(ASCII)	‘0’,‘1’,…,‘9’	‘A’,‘B’,…,‘Z’	‘＜’
编码(DEC)	0,1,…,9	10,11,…,35	0

每个信息字段分别计算,产生“校验 1”到“校验 4”。计算步骤如下:

(1) 字段中字符编码,生成 0～35 的 10 进制数据,然后循环乘以 7、3、1…的权重;

(2) 将第一步中的乘积加和;

(3) 将第二步中的加和对 10 求余;

(4) 余数(0～9)为校验码。

例如:证件号码为“HA672242＜”,计算过程如图 5-11 所示,最后校验码 1 为 6。

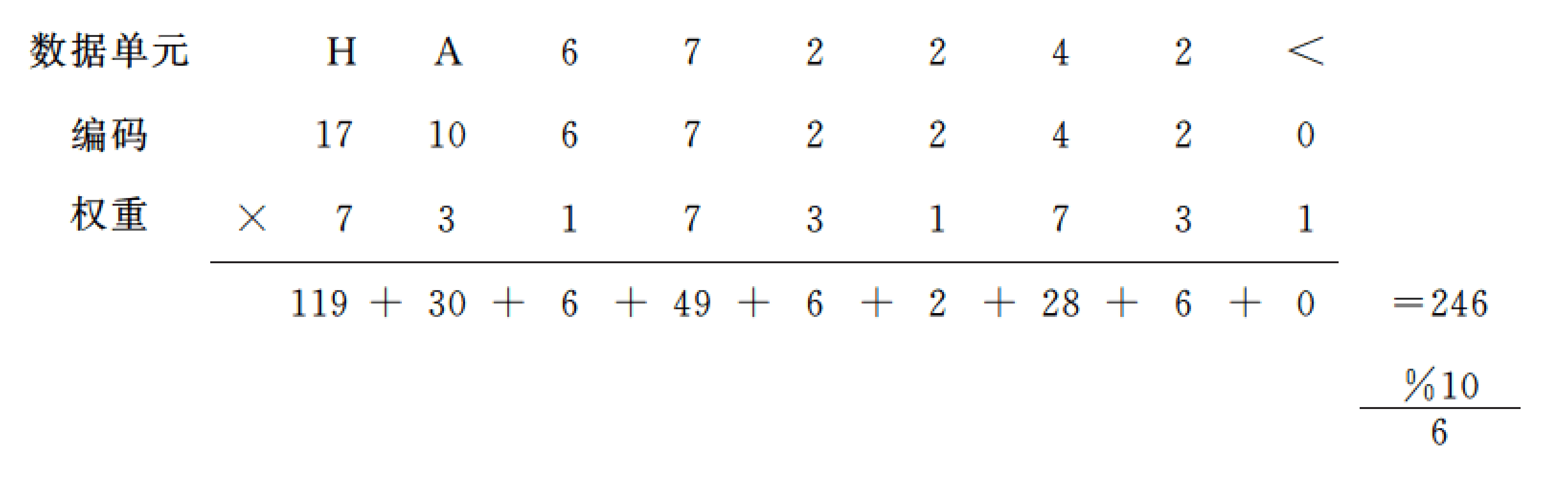


图 5-11 MRZ 校验码生成例子

其他校验码和总校验码的计算过程同上。

用 MRZ 部分信息生成密钥种子 K_{seed},然后从该 K_{seed} 计算出两个 3DES 密钥,如图 5-12 所示,分别用于建立文档基本访问控制密钥(K_{ENC} 和 K_{MAC})和安全消息的会话密钥。

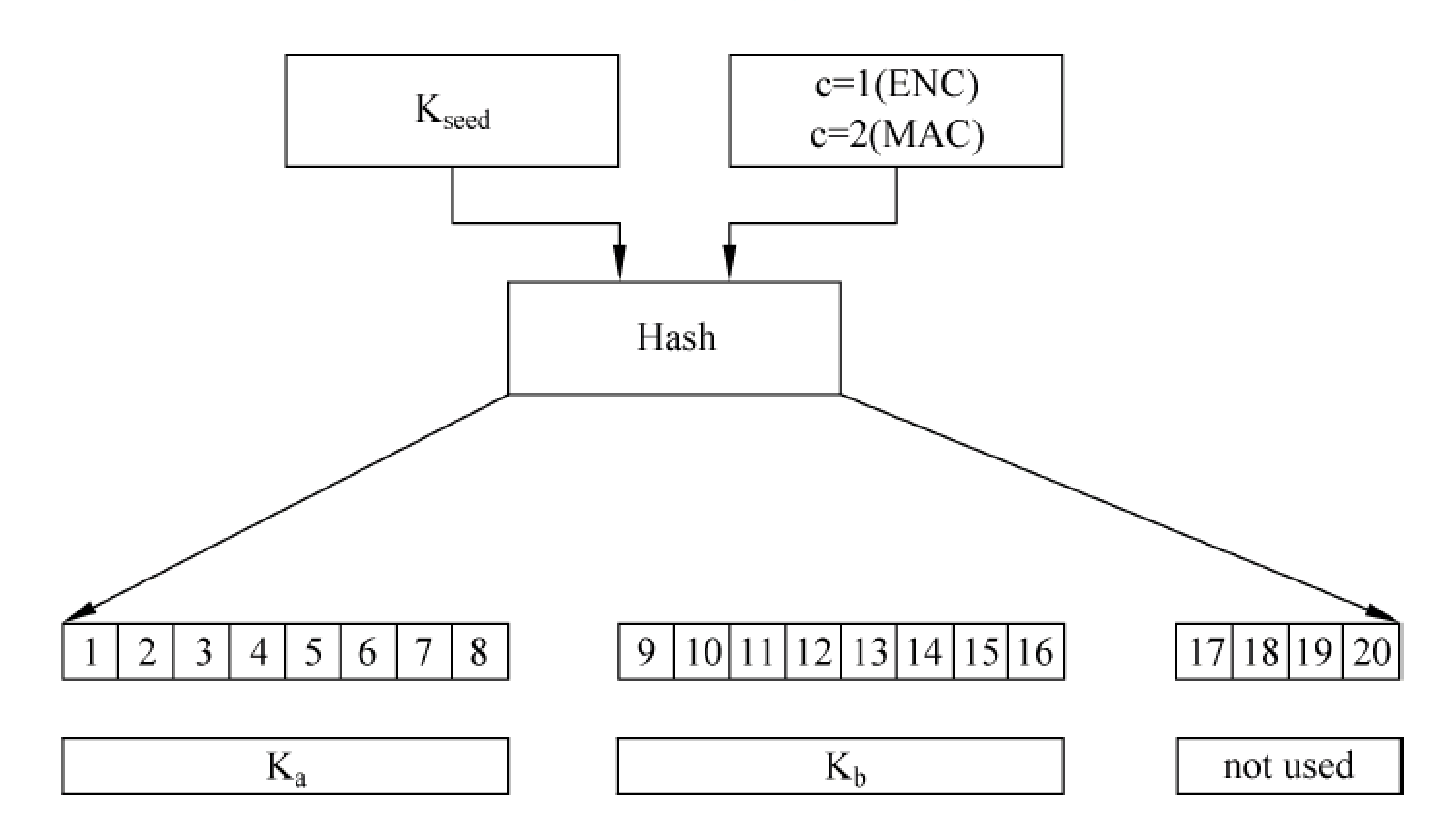


图 5-12 从密钥种子分散 3DES 密钥的方法

使用一个 32 位计数器,可以从单一种子密钥分散出多个密钥。根据密钥是用于加密或者 MAC 计算,下列值将被使用:

(1) c=1(0x00000001)用于加密；

(2) c=2(0x00000002)用于 MAC 计算。

从一个种子(K_{seed})和 c 中分散两个 3DES 密钥有如下步骤：

(1) 设 D 是 K_{seed} 和 c 的连接($D=K_{seed}||c$)；

(2) 计算 H = SHA1(D)，即 D 的 SHA1 散列值；

(3) 密钥 K_a=H 的第 1～8 个字节，密钥 K_b=H 的第 9～16 个字节；

(4) 调整密钥 K_a 和 K_b 的奇偶位，从而形成正确的 DES 密钥。

5.5.2 相互认证过程

电子旅行证件 DOC 9303 规范规定的相互认证的过程：

证件和机具分别根据使用 3DES 作为一个块加密算法(参见 ISO/IEC 11770-2 密钥建立机制 6)和三重询问-响应协议提供密码验证和建立功能。按照 ISO/IEC 9797-1 MAC 算法 3 计算密码校验和，然后附加到密文之后。交换的临时数据必须是 8 字节，交换的密钥素材必须是 16 字节，不需要使用不同的标识符。流程如图 5-13 所示。

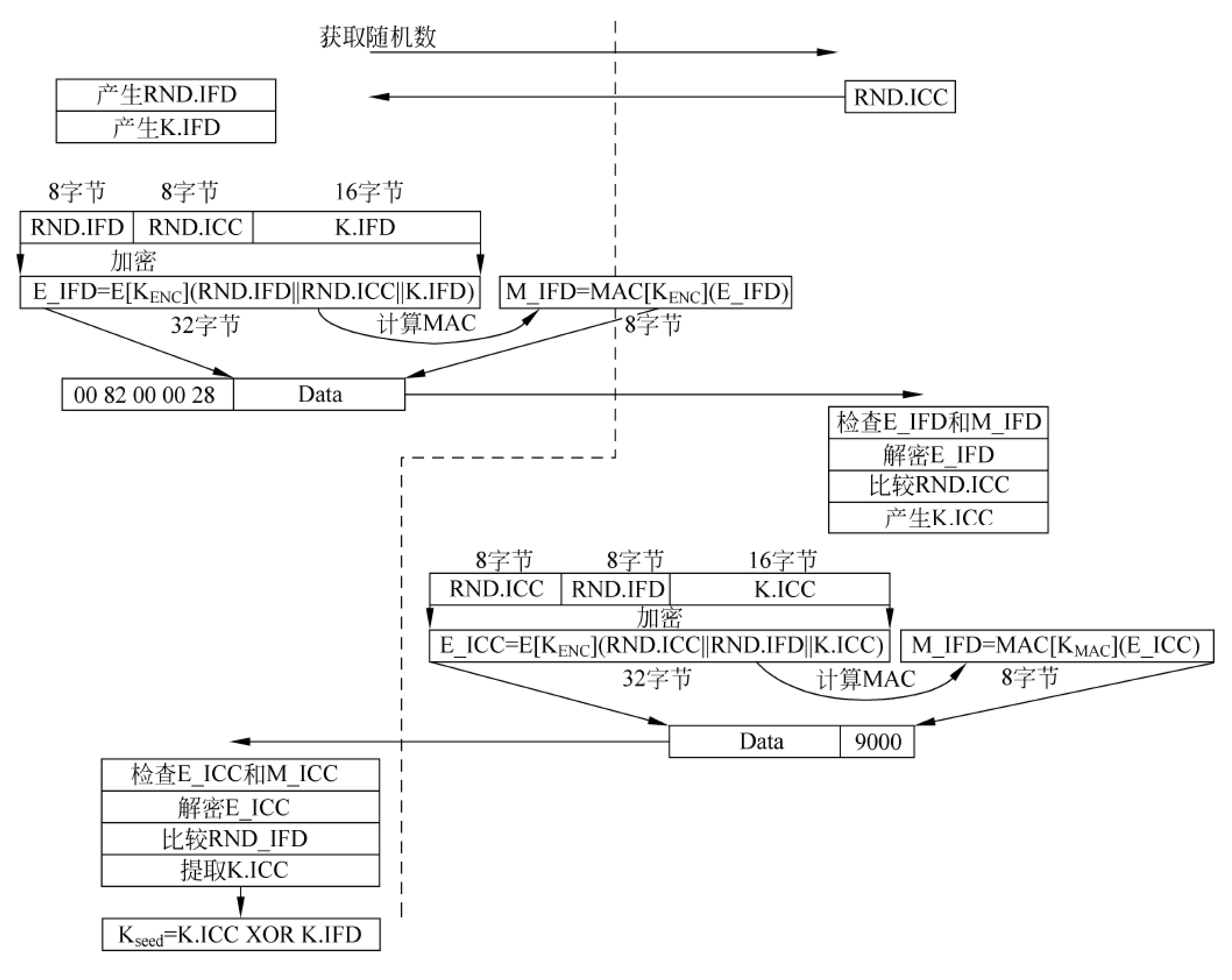


图 5-13 相互认证流程图

电子旅行证件 DOC 9303 规范相互认证的具体流程如表 5-4 描述。

表 5-4　相互认证流程

	机具(IFD)		卡(ICC)
1	发送 Get Challenge 命令	→	
	RND. ICC	←	返回响应(RND. ICC)
2	机具(IFD)完成以下的工作：		
	产生 RND. IFD 和 K. IFD	a	
	S=RND. IFD||RND. ICC||K. IFD	b	
	E_IFD=E[K_{ENC}](S)	c	
	M_IFD=MAC[K_{MAC}](E_IFD)	d	
	发送 Mutual Authenticate 命令(CLA=04,E_IFD||M_IFD 作为数据)	→	
3			卡(ICC)完成以下的操作：
		a	检查 E_IFD 的校验和 M_IFD
		b	解密 E_IFD
		c	从 S 中提取 RND. ICC,检查机具是否返回正确的值(与保存的 RND. ICC 比较)
		d	产生 K. ICC
		e	R=RND. ICC||RND. IFD||K. ICC
		f	E_ICC=E[K_{ENC}](R)
		g	M_ICC=MAC[K_{MAC}](E_ICC)
		←	返回响应(E_ICC||M_ICC)
4	机具(IFD)完成以下的操作：		
	检查 E_ICC 的校验和 M_ICC	a	
	解密 E_ICC	b	
	从 R 中提取 RND. IFD,检查卡是否返回正确的值(与保存的 RND. IFD 比较)	c	

完成上述操作后,机具和卡拥有相同的会话密钥：

K_{seed_s}=K. ICC ⊕ K. IFD

SSC=RND. ICC 的最后 4 个字节||RND. IFD 的最后 4 个字节

KS_{ENC}=SHA1 (K_{seed_s}||00000001)

KS_{MAC}=SHA1 (K_{seed_s}||00000002)

注：

- 在返回 Mutual Authenticate 命令的响应之前,完成会话密钥的计算；
- RND. ICC,RND. IFD,M_IFD,M_ICC 的长度是 8 个字节；
- K. IFD,K. ICC 的长度是 16 个字节；
- 计算 E_IFD 和 E_ICC 的算法是 DES CBC；
- ||表示连接操作；⊕表示异或操作。

5.6　扩展访问控制

在 2004 年 12 月 31 日欧盟委员会 2252/2004 号条例规定了电子护照和旅行证件的安全特征及生物识别标准之后,欧盟颁布了针对其所有成员国推行电子护照的规定。2006 年 8 月是最迟日期,这些电子护照要求将护照持有者的数码照片连同其他资料,如姓名、出生日期及

国籍等都存储在旅行证件内的一枚芯片上。这些电子数据由基本访问控制(basic access control,BAC)的安全协议保护并通过被动认证(passive authentication,PA)保证数据的完整性。

2006年6月,欧盟委员会发布新条例,规定在欧盟成员国的护照上增加两枚指纹作为附加生物识别信息,要求最晚于2009年6月全面施行。德国在2008年11月进行了系统迁移,是欧洲第一个迁移到新系统的国家。2008年9月欧盟成员国在奥地利的布拉格举行了EAC 1.11版本的互操作性测试。随后德国BSI组织发布了EAC 2.0规范。在EAC研究方面,欧洲已经走在世界前列。当然新加坡也提出了自己的扩展访问控制规范。

通常具有EAC安全机制的电子护照读写流程如图5-14所示。

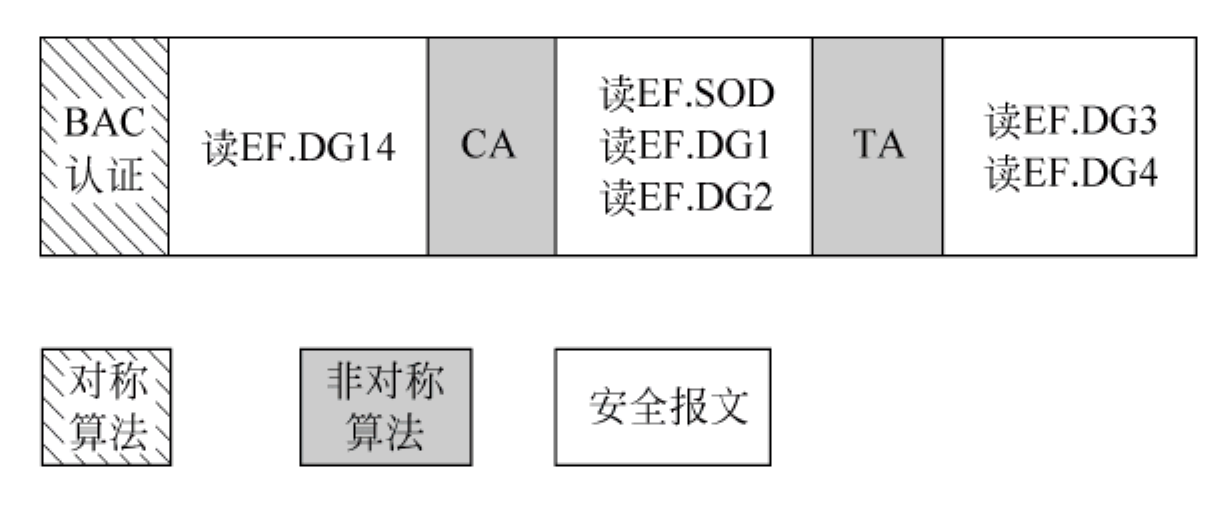


图5-14 支持EAC的护照数据读取流程

EAC访问控制分为两个环节:芯片认证(chip authentication,CA)和终端认证(terminal authentication,TA)两部分,其中芯片认证(CA)可作为独立协议代替主动认证(active authentication,AA),而终端认证(TA)只能和芯片认证组合使用,TA认证机具终端是否有授权读取护照持证人的隐私数据(指纹和虹膜)。芯片认证去掉了挑战应答机制,而是采用密钥交换协议(比如DH和ECDH)进行机具终端和护照的密钥交换,根据受国家签发证书保护的EF.DG14(含公钥)和具有的私钥进行匹配,防止克隆。又通过密钥交换进行协议新的密钥种子,进行BAC方式的读取。芯片认证和终端认证可以使用不同的算法,相应的算法组合如表5-5所示。

表5-5 根据算法不同CA和TA组合表

序号	CA	TA	备　　注	性能
1	DH	RSA	占内存小,算法较慢	优
2	DH	ECDSA	占内存大	差
3	ECDH	RSA	占内存大	最差
4	ECDH	ECDSA	占内存小,算法快	最优

2008年9月欧盟成员国在奥地利的布拉格举行了EAC 1.11版本的互操作性测试,其算法统计数据表明:选用组合(1)的护照样本占10/34,选用组合(2)的占2/34,选择组合(3)的占0/34,选择组合(4)的占22/34。

本书将以最优的算法组合ECDH+ECDSA进行EAC流程和安全机制剖析。

5.6.1 CA流程

AA认证是为了防止电子数据的恶意克隆,它采用挑战应答机制进行。机具终端发送一个挑战值(随机数或者某个有意义的串)给护照,护照对这个挑战利用AA私钥进行签名,

回送给终端。终端再用护照的 AA 公钥进行验证。这种机制存在一个缺陷,如果机具终端发送的挑战具有特殊意义,比如:$c=\mathrm{Sign}(SK_{PCD},ID_{PICC}||Date||Time||Location)$,此时这个挑战发送给护照,护照签名后回送。所有具有机具终端公钥的机构都可以根据这个受到护照签名、机具签名的挑战值,来确认护照持证人在某个时间某个地点出现过,该持证人的行踪隐私则被暴露。

芯片认证(CA)主要完成对卡片真伪的认证,主要根据公钥机制中公私密钥成对匹配的原理实现。CA 认证不再使用挑战应答机制,而是采用密钥交换协议进行隐式密钥对的匹配验证。机具终端通过读取存有交换算法和卡片公钥的文件 DG14,再自行产生相同算法的公私密钥对。卡片认证的流程如图 5-15 所示。

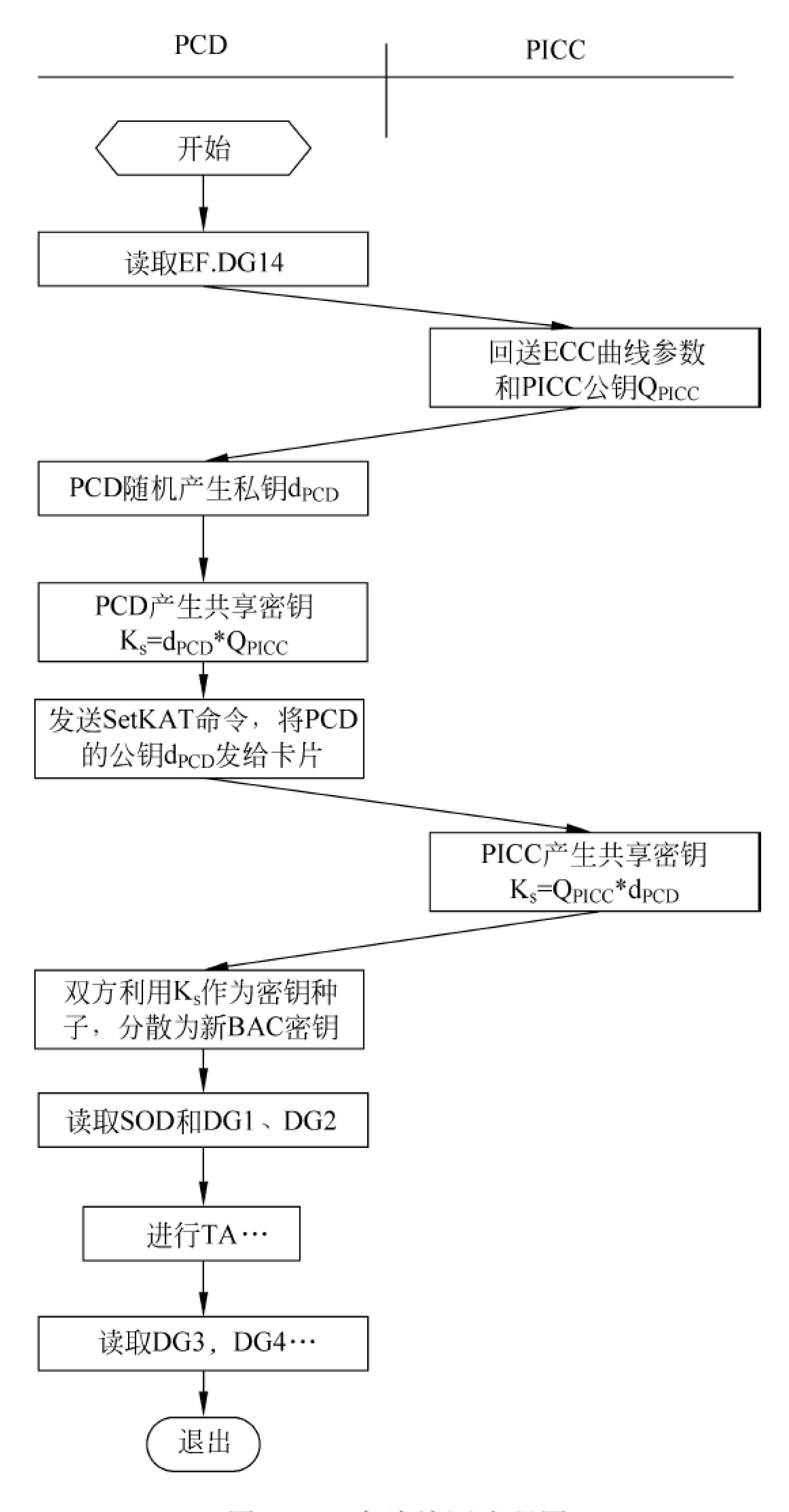


图 5-15 卡片认证流程图

5.6.2 TA 流程

终端认证过程(TA)主要完成对终端是否具有授权的认证。这个过程主要是对三个证书(CVCA 国家证书、DVCA 护照签发证书、IS 终端证书)的验证过程。这三个证书的证书链是：IS 终端证书受 DVCA 签名、DVCA 护照证书受 CVCA 证书签名。DVCA 证书是通过外交途径发往护照发行国进行 CVCA 签名的。证书验证过程是根据 ECDSA 算法的验证过程，具体验证流程如图 5-16 所示。

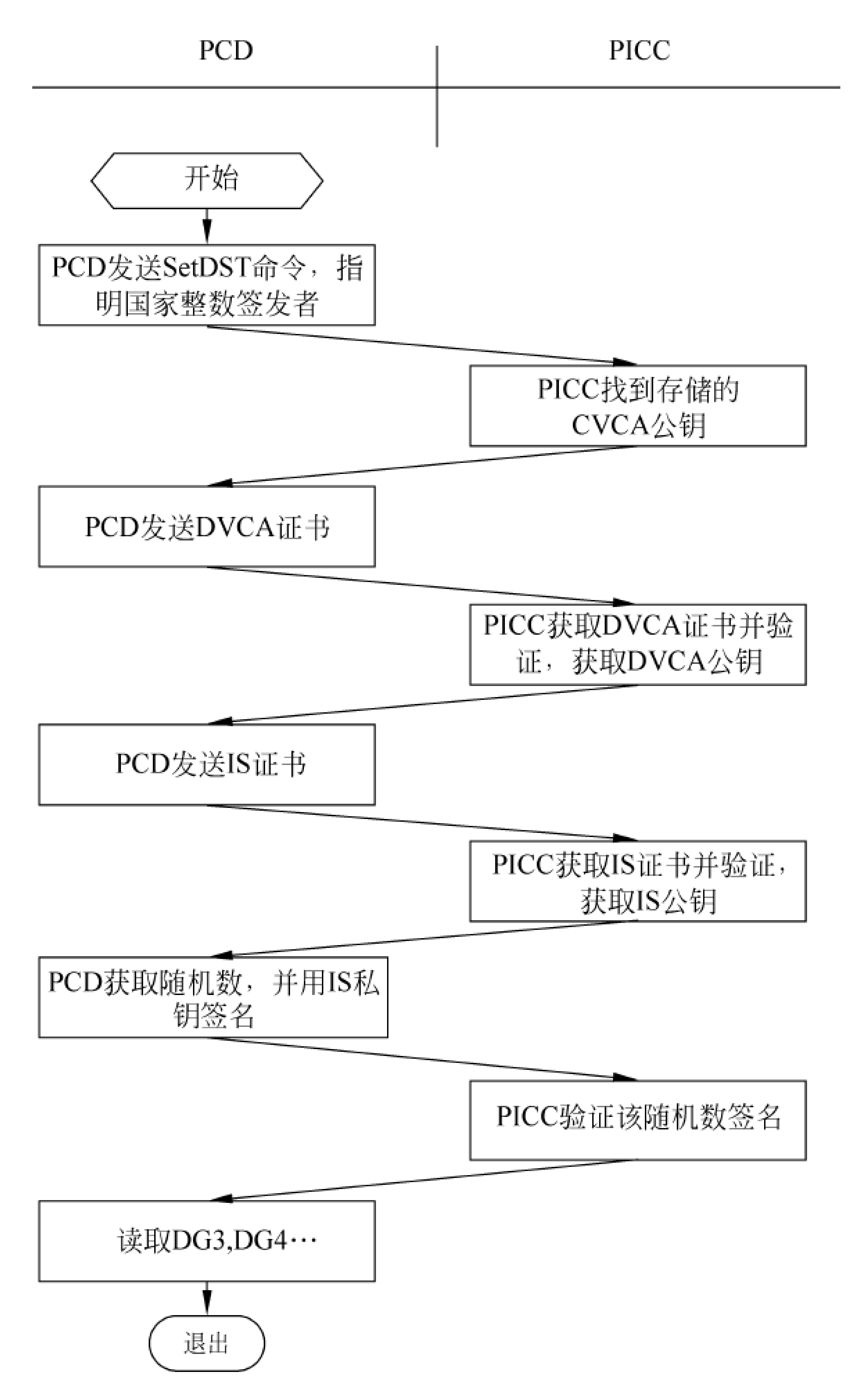


图 5-16　终端认证流程图

边检海关如果需要读取护照持证人的指纹或虹膜等隐私问题，则必须通过 TA 认证。

5.6.3 密钥交换

在 CA 过程中，需要 ECDH 密钥交换算法，ECDH 算法流程如下描述。终端 A 将要发送的消息给护照 B，两者首先协商一组椭圆曲线参数六元偶，然后 A 和 B 要做的是：

- A 随机选取整数 $1<k_A<n-2$，计算 $Q_A=k_AG$，并将 Q_A 发送给 B；

- B 随机选取整数 $1<k_B<n-2$，计算 $Q_B=k_BG$，并将 Q_B 发送给 A；
- A 收到 B 发来的 Q_B 后计算 $k_AQ_B=k_Ak_BG$；
- B 收到 A 发来的 Q_A 后计算 $k_BQ_A=k_Bk_AG$。

由于 $Q=k_Ak_BG=k_Bk_AG$，于是机具终端 A 和护照 B 共同拥有的会话密钥为 Q。

EF.DG14 存储椭圆曲线的参数，图 5-17 显示了 EF.DG14 的文件结构，文件头部分指明密钥交换算法和 OID，后面紧跟椭圆曲线参数 p、a、b、G、n 和公钥 Q。

```
01 26 30 82 01 22 06 09 04 00 7f 00 07 02 02 01   OID=0.4.0.127.0.7.2.2.1.2
02 30 82 01 13 30 81 d4 06 07 2a 86 48 ce 3d 02   ECDH
01 30 81 c8 02 01 01 30 28 06 07 2a 86 48 ce 3d   OID=1.2.840.10045.1.1
01 01 02 1d 00 d7 c1 34 aa 26 43 66 86 2a 18 30   素数 p
25 75 d1 d7 87 b0 9f 07 57 97 da 89 f5 7e c8 c0
ff 30 3c 04 1c 68 a5 e6 2c a9 ce 6c 1c 29 98 03   参数 a
a6 c1 53 0b 51 4e 18 2a d8 b0 04 2a 59 ca d2 9f
43 04 1c 25 80 f6 3e cf e4 41 38 87 07 13 b1 a9   参数 b
23 69 e3 3e 21 35 d2 66 db b3 72 38 6c 40 0b 04
39 04 0d 90 29 ad 2c 7e 5c f4 34 08 23 b2 a8 7d   参数 G
c6 8c 9e 4c e3 17 4c 1c 6e fd ee 13 c0 7d 58 aa
56 f7 72 c0 72 6f 24 c6 b8 9e 4e cd ac 24 35 4b
9e 99 ca a3 f6 d3 76 14 02 cd 02 1d 00 d7 c1 34   参数 n
aa 26 43 66 86 2a 18 30 25 75 d0 fb 98 d1 16 bc
4b 6d de bc a3 a5 a7 93 9f 02 01 01 03 3a 00 04
b3 c2 1a 5f ee 96 32 49 1e 8e da 4d 66 ea 5a f5
29 64 80 99 3d eb 9d cd 58 63 39 fd 0a f7 60 e7   公钥 Q 的坐标 x，y
70 ac 41 fe ef d6 84 38 87 18 fe 07 aa 47 c0 ae
07 cd f1 b3 b0 08 45 1e
```

图 5-17 EF.DG14 文件结构图

PKI 的发展与密钥理论的发展密不可分。在 EAC 机制中，终端认证过程主要是依赖于 PKI 完成对终端的授权鉴别过程，使用 PKI 是必需的。证书发行时需要利用国家签名证书中心(Country Signing Certificate Authority，CSCA)签发证件签名者证书，利用该证书对各 DGx 的哈希值进行签名，将证件签名者证书和签名值一起放在 EF.SOD 文件中，供被动认证时验证。具体流程如图 5-18 所示。

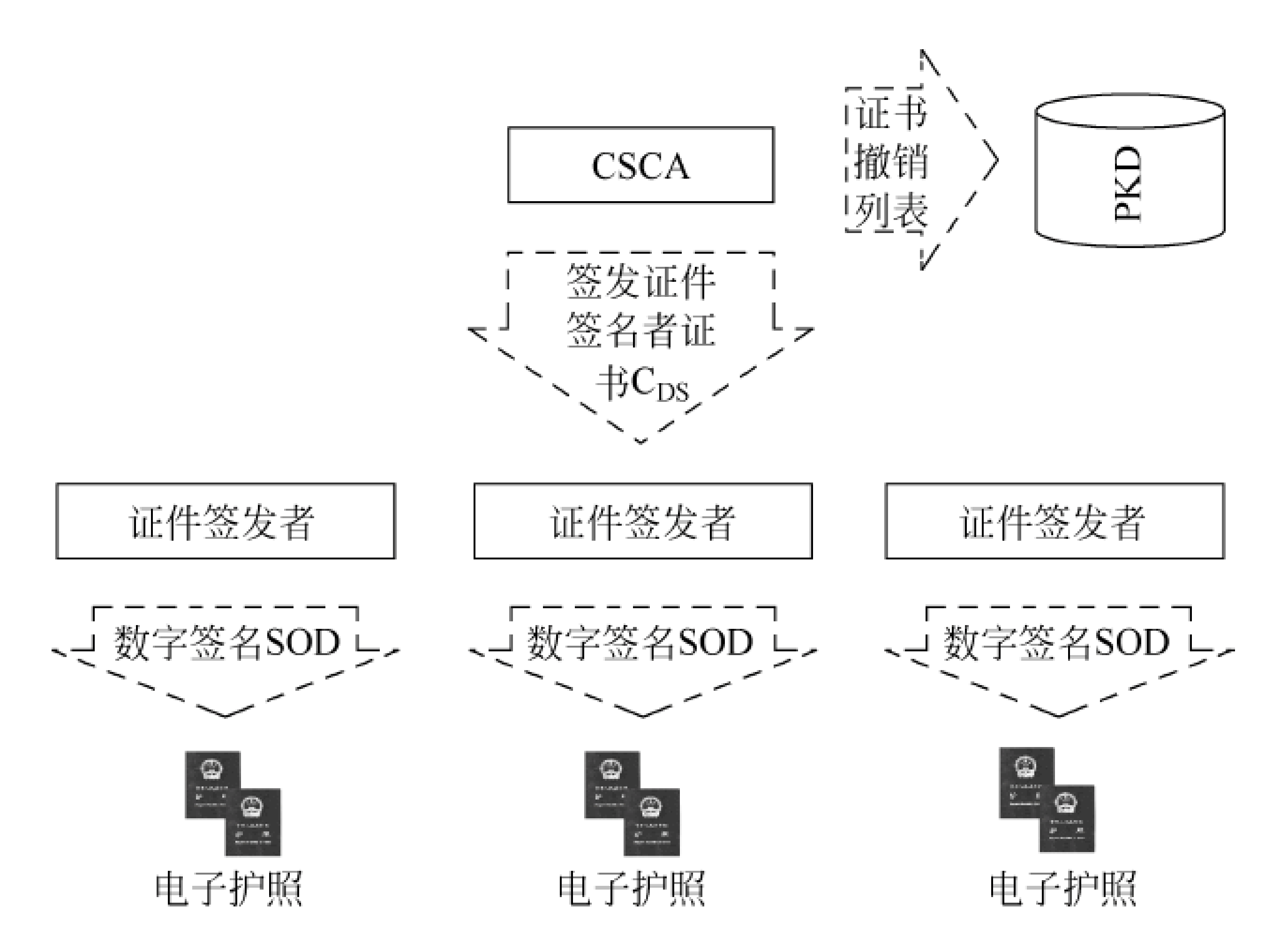


图 5-18 EAC 机制中 PKI 证书签发结构

护照发行国和海关边检所在国必须通过外交途径，利用国家验证证书中心(Country Verification Certificate Authority，CVCA)对证件验证中心(Document Verifier Certificate Authority，DVCA)进行证书的颁发。DVCA 对边检海关的机具终端(inspection system，IS)颁发证书。证书按照级别从上而下验证，具体的验证流程如图 5-19 所示。

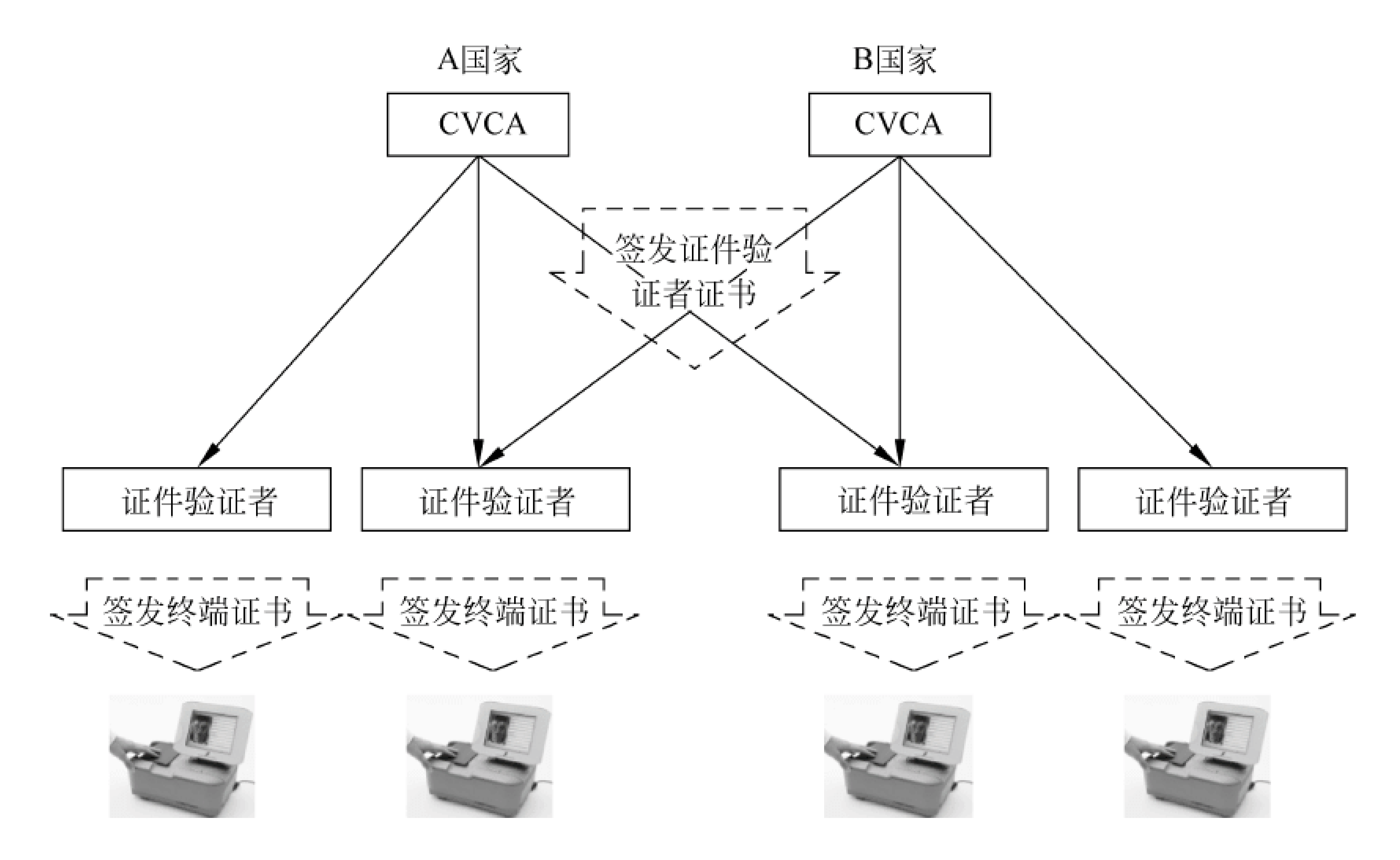


图 5-19 EAC 机制中 PKI 证书验证结构

首先，EAC 机制执行 BAC 协议，并通过在非对称密钥对上运行最新版本的 PKI 检查芯片和检验系统的真实性。其次，对于检验而言，各国边境管制区域的系统都要求配备护照签发国的证书文件作为存储在芯片上的指纹和虹膜的安全存取的前提条件。EAC 还支持通过在数据传输过程中建立强大的对话密钥，来实现最为安全的数据编码。

5.7 安全报文

5.7.1 传输方式

根据安全性能设计的需要，智能卡与读写机具之间通常有以下 4 种报文传输方式。

(1) 明文传输

当对数据传输时的私密性、完整性、可靠性没有要求时，可以采用明文传输的方式。采用明文传输命令的数据域和响应的数据域均是明文，不需要做数据变换。

(2) 密文传输

当只对数据传输时的私密性有要求时，尽量采用密文传输的方式。密文传输可以使数据被安全算法加密，防止未经授权的第三方获取数据信息。

(3) 明文加校验传输

当只对数据传输时的完整性、可靠性有要求时，尽量采用 MAC 传输的方式。MAC 码传输可以保证数据传输时不被篡改。MAC 码是使用命令的所有元素(包括命令头和命令

数据域中的数据)来产生的。以保证命令连同数据能够正确完整地传送,并对发送方进行认证。

(4) 密文加校验传输

当对数据传输时的私密性、完整性、可靠性均有要求时,采用密文 MAC 传输的方式。

后 3 种属于安全报文传输,密文传输可以保证数据的私密性,MAC 传输可以保证数据传输的完整性和可靠性。三者之间的比较如表 5-6 所示。

表 5-6 安全报文方式比较

报文传输方式 / 属性比较	密文传输	明文加校验	密文加校验
密文密钥	对称密钥	无	对称密钥
MAC 密钥	无	对称密钥	对称密钥
安全性	中	中	高
复杂性	中	低	高
特点	保护数据在传输过程中的私密性	保护数据在传输过程中的完整性	保护数据在传输过程中的私密性、完整性

5.7.2 安全消息计算

根据 ISO/IEC 7816-4 的规定,使用安全消息对 APDU 命令-响应进行保护的方法如图 5-20～图 5-23 所示。

按照 CASE1～CASE4 分类计算:

(1) CASE1 命令和响应安全消息格式如图 5-20 所示。

CASE1 APDU命令-响应

command header	command body
CLA INS P1 P2	

response body	response trailer
	SW1-SW2

CASE1 安全的APDU命令-响应

command header	command body		
CLA INS P1 P2	new Lc	T L CC	new Le='00'

{CLA INS P1 P2 padding}

response body		response trailer
T L SW1-SW2	T L CC	SW1-SW2

{T L SW1-SW2 padding}

new Le=0x0a	T=0x8e, L=0x08	T=0x99, L=0x02	T=0x8e, L=0x08

图 5-20 安全信息:APDU 命令-响应 CASE1

（2）CASE2 命令和响应安全消息格式如图 5-21 所示。

CASE2 APDU命令-响应

command header	command body
CLA INS P1 P2	Le field

response body	response trailer
data field	SW1-SW2

CASE2 安全的APDU命令-响应

command header	command body			
CLA INS P1 P2	new Lc	T L Le	T L CC	new Le='00'

{CLA INS P1 P2 padding}{T L Le padding}

response body			response trailer
T L Value	T L SW1-SW2	T L CC	SW1-SW2

{T LValue T L SW1-SW2 padding}

T=0x09	T=0x8e, L=0x08	T=0x99, L=0x02	T=0x8e, L=0x08	T=0x81 or L 0x87

图 5-21　安全信息：APDU 命令-响应 CASE2

（3）CASE3 命令和响应安全消息格式如图 5-22 所示。

CASE3 APDU命令-响应

command header	command body	
CLA INS P1 P2	Le field	data field

response body	response trailer
	SW1-SW2

CASE3 安全的APDU命令-响应

command header	command body			
CLA INS P1 P2	new Lc	T L Value	T L CC	new Le='00'

{CLA INS P1 P2 padding}{T L Value padding}

response body		response trailer
T L SW1-SW2	T L CC	SW1-SW2

{T L SW1-SW2 padding}

T=0x8e, L=0x08	T=0x99, L=0x02	T=0x8e, L=0x08	T=0x81 or L 0x87

图 5-22　安全信息：APDU 命令-响应 CASE3

（4）CASE4 命令和响应安全消息格式如图 5-23 所示。

5.7.3　范例

在读写机具和电子护照成功执行认证协议后，读写机具和护照芯片都根据 BAC 模式中描述的密钥分散机制，用 K. ICC 异或 K. IFD 作为密钥种子，计算会话密钥 KS_{ENC} 以及

CASE4 APDU命令-响应

command header	command body		
CLA INS P1 P2	Lc field	data field	le field

response body	response trailer
data field	SW1-SW2

CASE4 安全的APDU命令-响应

command header	command body					
CLA INS P1 P2	new Lc	T L Value	T L Le	T L CC	new Le='00'	{CLA INS P1 P2 padding}{T L Value T L Le padding}

response body			response trailer	
T L Value	T L SW1-SW2	T L CC	SW1-SW2	{T LValue T L SW1-SW2 padding}

T=0x97	T=0x8e, L=0x08	T=0x99, L=0x02	T=0x8e, L=0x08	T=0x81 or L 0x87

图 5-23 安全信息：APDU 命令-响应 CASE4

KS_{MAC}。所有后续的通信必须在具有 MAC_ENC 模式的安全消息的保护下进行。

SM(secure messaging)数据对象必须根据表 5-7 所示，按照如下顺序来使用。

表 5-7 数据对象和 APDU

	用　途	命令 APDU	响应 APDU
DO‘87’	加密数据	如果传送数据，必须存在	如果传送数据，必须存在
DO‘81’	明文数据	如果传送数据，必须存在	如果传送数据，必须存在
DO‘97’	数据长度	如果需要数据，必须存在	不使用
DO‘99’	状态字(SW1-SW2)	不使用	必须存在，除非出现 SM 错误
DO‘8E’	密文校验和(MAC)	必须存在	如果 DO‘87’(DO‘81’)和/或 DO‘99’存在，则必须存在

注 1：DO‘81’和 DO‘87’仅可以存在一种

注 2：在 DO‘87’中，indicator 字段＝0x01 表示采用 3DES-CBC 模式(ICAO 规定的模式)对数据进行加密；＝0x81 表示采用 3DES-ECB 模式对数据进行加密。

命令 APDU：[DO‘87’]([DO‘81’]) [DO‘97’] DO‘8E’；

响应 APDU：[DO’87’]([DO‘81’]) DO‘99’ DO‘8E’。

所有 SM 数据对象都必须按照 ISO/IEC 7816-4 规范的 BER TLV 来编码。指令头必须包含在 MAC 计算中，因此必须使用类型字节 CLA ＝0x0c。

Lc 的实际值在使用安全消息后调整为 Lc’。如果需要，可选择一个适当的数据对象包含在 APDU 数据部分中，来传递 Lc 的原值。在受保护的命令 APDU 中，新 Le 必须设置为‘00’。

命令 APDU 和响应 APDU 的安全报文封装过程如图 5-24 和图 5-25 所示。

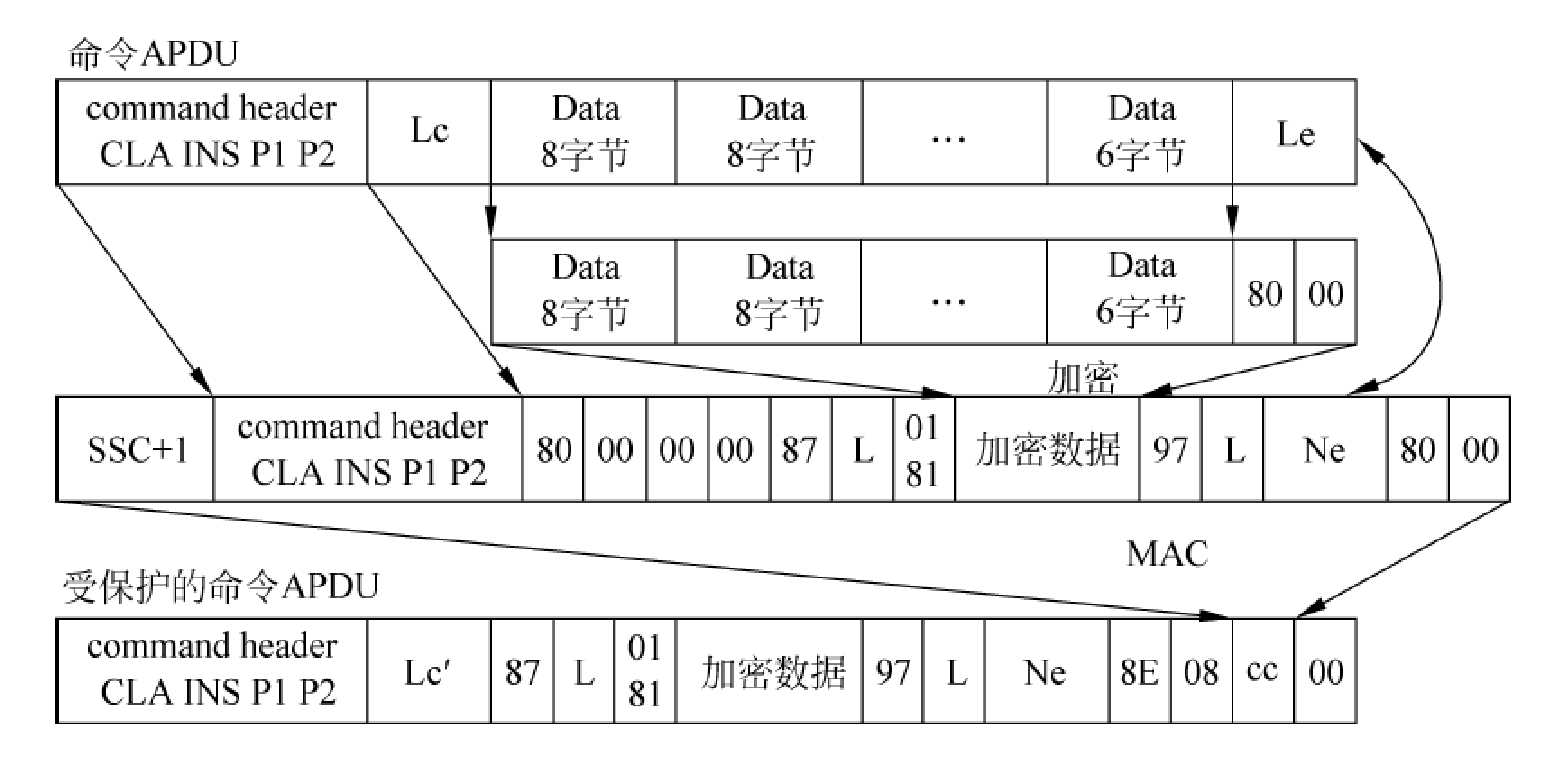


图 5-24　安全命令封装过程

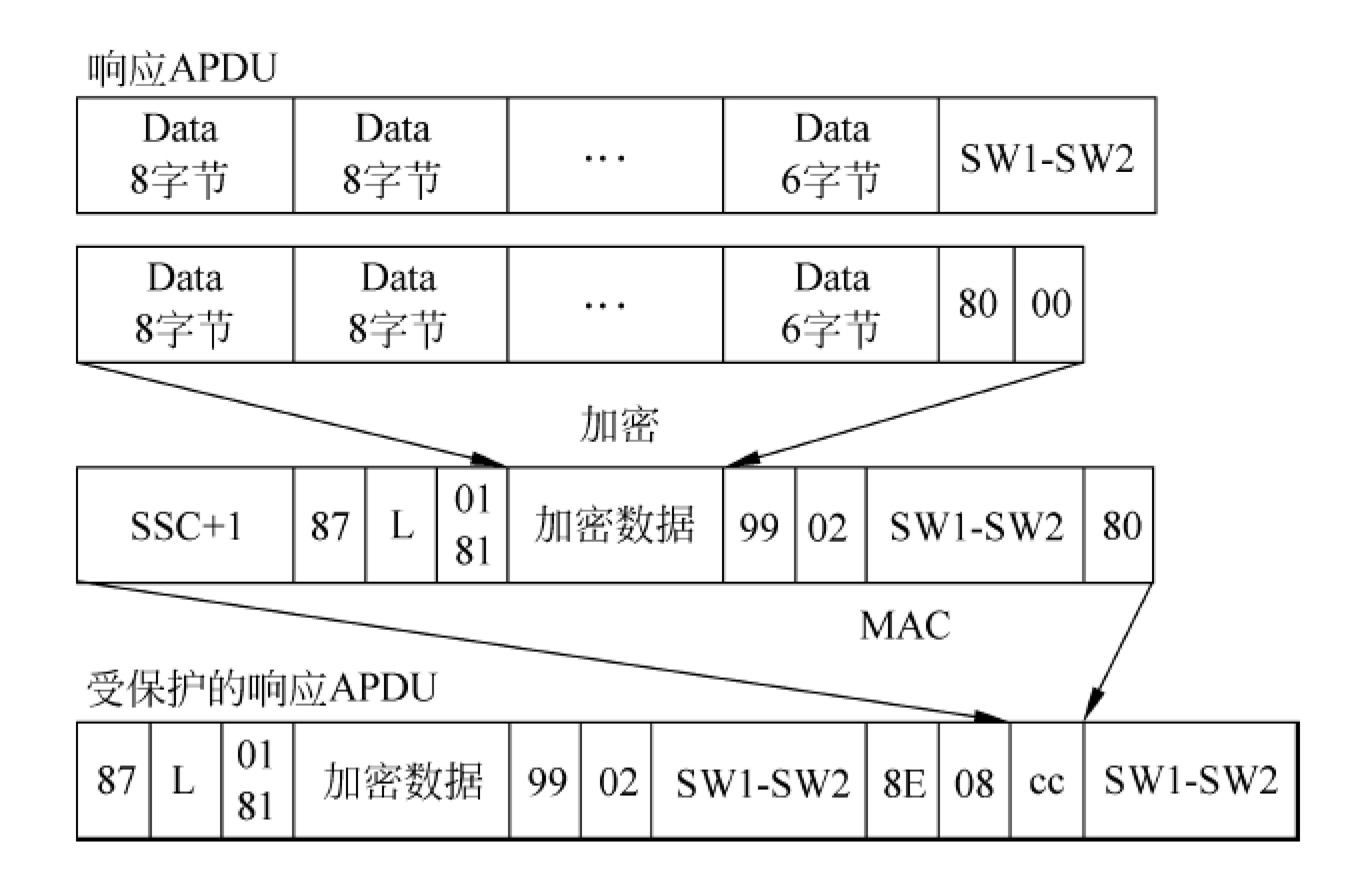


图 5-25　安全响应封装过程

5.8　数字签名

数字签名在身份认证、数据完整性、不可否认性和匿名性等信息安全领域有着重要作用。在现实生活中，通常采用手写签名、印章和封印等手段便可以获得在法律上认可的身份鉴别、数据完整性认证和抗否认效果。在数字化活动时，迫切需要一种能够进行身份鉴别、数据完整性认证和抗否认的技术，数字签名(digital signature)技术应运而生。

ISO对数字签名是这样定义的：附加在数据单元上的一些数据，或是对数据单元所做的密码变换，这种数据或变换允许数据单元的接收者用以确认数据单元来源和数据单元的完整性，并保护数据，防止被人(如接收者)伪造。

数字签名是以密码学的方法对数据文件产生的一组代表签名者身份和数据完整性的数据信息。它提供了一种鉴别方法，以解决伪造、抵赖、冒充等问题。数字签名在信息安全中包括身份认证、数据完整性、不可否认性以及匿名性等方面的重要应用。在法定证件应用中，数字签名在保护个人信息、防止证件伪造或变造上起着重要的作用。

5.8.1 算法分类

数字签名按使用的加密算法可分成两种方式，分别为基于对称密钥算法的数字签名和基于非对称密钥算法的数字签名。

用对称密钥算法实现数字签名必须有仲裁人参与。假设用户 A 和仲裁人 T 共享密钥 K_A，用户 B 和 T 共享另一个不同的密钥 K_B。对称密钥数字签名的基本思想是：

(1) A 用 K_A 加密明文消息 M 和身份证明 t，并把加密消息 C_A 传送给 T：$C_A = E_{KA}(M,t)$。

(2) T 用 K_A 解密 C_A 得到明文：$(M, t) = D_{KA}(C_A)$。

(3) T 用 K_B 加密成 C_B：$C_B = E_{KB}(M,t)$。

(4) T 把加密的消息包 C_B 连同 A 用 K_A 加密的 C_A 一起传给 B：$(C_B,C_A) \rightarrow B$。

(5) B 用 K_B 解密消息包 C_B 之后，就可以读到 A 的消息 M 和 T 的签名证书 t，证明消息来自 A：$(M,t) = D_{KB}(C_B)$。B 需要保留 C_A 以备发生分歧时裁决之用。这种签名方式要求仲裁人必须高度的完善和安全，而且得到所有人的信任，如图 5-26 所示。

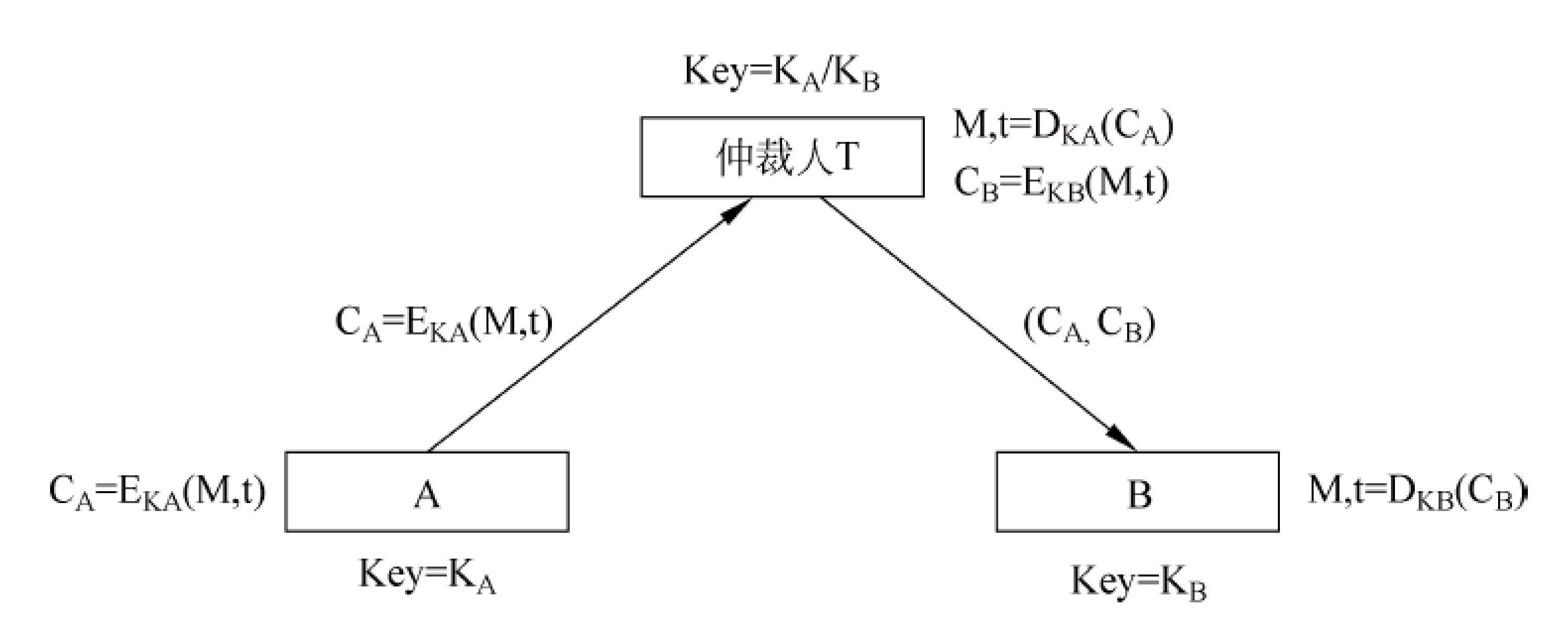


图 5-26 基于对称密钥算法的数字签名

在公钥密码学中，密钥是由公开密钥和私有密钥组成的密钥对。数字签名就是用私有密钥进行加密，接收方用公开密钥进行解密。由于公开密钥不能推算出私有密钥，所有公开密钥不会损害私有密钥的安全；公开密钥无需保密，可以公开传播，而私有密钥必须保密。因此，当某人用私有密钥加密信息，能够用他的公开密钥正确解密，就可肯定该消息是某人签字的，这就是数字签名的基本原理。因为其他人的公开密钥不可能正确解密该加密过的消息，其他人也不可能拥有该人的私有密钥而制造出该加密的消息。

在基于非对称密钥算法的数字签名应用中，原始信息经过对称密钥加密，进行数据的私密性保护。将消息摘要进行私钥签名，生成签名信息，附加在加密信息后一起发送到接收方。接收方进行签名校验和信息的解密，比较摘要。具体工作过程如图 5-27 所示。

5.8.2 ECDSA 签名

椭圆曲线密码系统与现存的公钥密码体制具有相同的安全性，但密钥长度却相对要短，较短的密钥意味着较快的运算和较小的内存要求，在智能卡 COS 中，基于 ECC 的签名算法将越来越多地被应用。

一般可以将 DLP 上的签名机制平行移植到 ECDLP 上，从而获得 ECC 签名机制。较具影响的 ECC 签名机制有 ECDSA 签名机制；CNR 签名机制；EC-ElGamal 签名机制；不带有消息恢复功能的椭圆曲线数字签名机制，即验证方需要用消息原文才能验证签名的有效

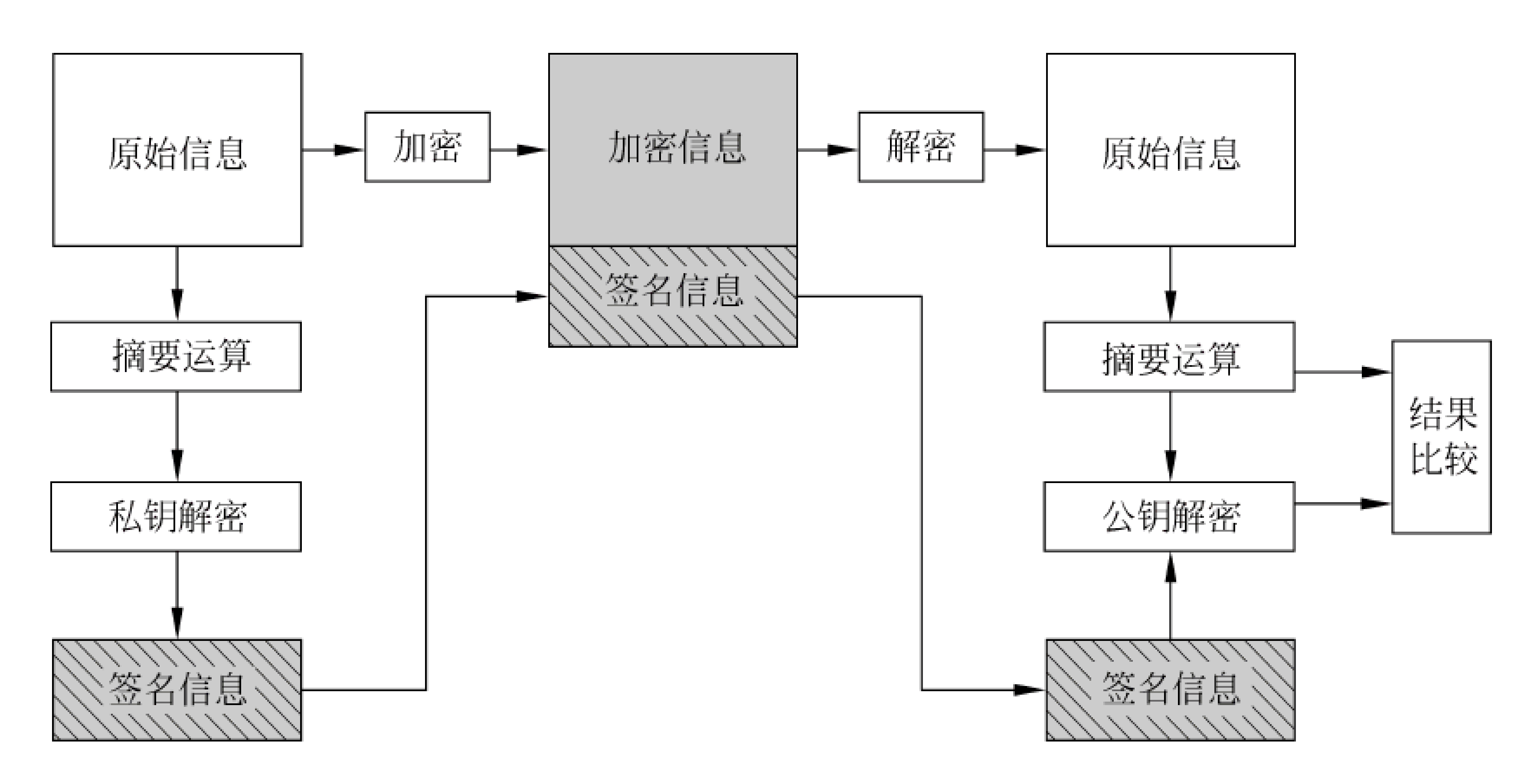


图 5-27 基于非对称密钥算法的数字签名

性；带有消息恢复功能的椭圆曲线数字签名机制，即需要从消息中恢复出来原文才能进行下一步的签名验证。

ECDSA(elliptic curve digital signature algorithm)是 Elgamal 椭圆曲线数字签名的改版，也非常地类似 DSA 数字签名，不带有消息恢复功能。首先要给出椭圆曲线域参数，以精确定义一条椭圆曲线和一个基点，进而确定曲线上的各点运算。在 SEC1 和 P1363 ECC 工作草案中，定义椭圆曲线域参数 T 为 $T=(F\ a\ b\ P\ m\ h)$，其中，F 表示有限域 $GF(2^n)$，a，$b\in GF(2^n)$，P 表示一个基点，m 为素数，是基点 P 的阶，余因子 $h=\#E(GF(2^n))/m$，$\#E(GF(2^n))$为椭圆曲线的阶。举例，点 Alice 用 ECDSA 算法对消息 M 进行签名并发送被 Bob。

1. 签名过程

(1) 计算 $e=H(M)$，Hash 为单向散列函数；

(2) 选取整数 $k(1<k<n-1)$，计算点$(x,y)=kP$，并设 $r=x \bmod m$；

(3) 利用 Alice 的私钥 d，计算 $s=k^{-1}(e+rd) \bmod m$；

(4) A 送给 Bob 消息 M 及签名(r,s)。

2. 认证签名过程

(1) 首先查找 Alice 的公钥 Q，如果 $r \bmod m=0$ 则签名无效；

(2) 计算散列值 $e=H(M)$以及 $s^{-1} \bmod m$；

(3) 计算 $u=s^{-1}e \bmod m$ 和 $v=s^{-1}r \bmod m$；

(4) 计算点$(x_1,y_1)=uP+vQ$，当 $x_1 \bmod m=r$，Bob 接受 Alice 对消息 M 的签名。

5.8.3 EC-ElGamal 签名

ElGamal 具有多种数字签名算法，但不是所有的形如$(m,(r,s))$的数学组合都能产生安全的数字签名，文献给出了安全数字签名方案的设计规则，并列出了所有符合这种设计规则的 ElGamal 型签名方程和验证方程，将 ElGamal 数字签名体制移植到椭圆曲线密码系统上，首先要做的是私钥、公钥和参数 R 的相互映射，如表 5-8 所示。

表 5-8 ElGamal 密码体制与椭圆密码体制的映射关系

	ElGamal 密码体制	椭圆曲线密码体制
私钥	x	x
公钥	$y=g^x$	$y=xP$
R 的计算	$r=g^k$	$R=kP$

在椭圆曲线密码体制上实现 EC-ElGamal 数字签名方案的步骤主要包括以下内容：

- 系统初始化。
- 密钥的生成。
- 签名的生成。
- 签名体制的正确性检验。

ElGamal 数字签名是基于离散对数群的签名体制，我们将其应用到椭圆曲线密码体制上。系统参数包含：基于有限域的 $GF(p)$ 的椭圆曲线，有理点的个数 n 和基点 G，单向 Hash 函数，可以采用 FIPS 180-1（联邦信息处理标准）中的 Hash 算法，即 Hash：$\{0,1\}^* \to \{0,1\}^{160}$。举例，假设 Alice 的公钥为 Q，私钥为 d，Alice 希望对消息 m 进行签名并将签名对发送给 Bob，Bob 验证签名是否正确。

1. 签名过程

(1) 随即选择，$k\in[2,n-2]$，计算 $U=kG=(x_1,y_1)$；

(2) 计算 $r=x_1 \bmod n$，若 $r=0$，则返回 1；

(3) 利用 $sk=\text{Hash}(M)+dr \bmod n$ 计算 s，如果 $s=0$，则返回 1；

(4) (U,s) 即是 Alice 对消息 M 的签名，将 $(M,(U,s))$ 发送给 Bob。

2. 认证签名的过程

(1) 验证 $s\in[1,n-1]$，如果不是则拒绝签名；

(2) 验证 $sU=\text{Hash}(M)G+rQ$，如果等式成立，则接受签名，否则拒绝签名。

3. 签名验证的原理

由 $sk=\text{Hash}(M)+dr \bmod n$ 得

$$s=k^{-1}(\text{Hash}(M)+dr),$$

$$sU=k^{-1}(\text{Hash}(M)+dr)kG=\text{Hash}(M)G+drG=\text{Hash}(M)G+rQ$$

任何人如果没有 Alice 的私钥 d，不可能伪造签名。

5.8.4 计算范例

常用于数字签名的非对称密钥算法为 RSA 算法和 ECDSA 算法，按照其算法，分别计算出相应的数据如下所示，供读者参考。

1. RSA 算法签名

```
//消息原文
M =
      E2 0C 1C C0 69 7E CC F0 D2 88 72 98 45 2E F3 3E
      1A 6E E0 0A E0 0C DD 88 6A F2 07 60 69 7C BC 8A
      C9 42 98 EA B2 A6 15 AA 76 C0 7A 68 32 F0 95 90
      BF B6 3A D6 5E FA EB 42 28 92 6A C8 0A 0E 83 C3
```

```
        E6 7C 39 08 7A 1C 5A F8 37 44 F0 16 29 64 80 80
        BA 3C C9 02 68 A8 C1 B2 CD 2C 23 50 88 E6 55 6E
        22 1A 7B CA A4 0A E1 D0 B6 44 67 9A 6A 22 F4 8E
        D3 9A 6F D8 45 18 77 CE 54 A8 25 AC 4A E6 BB 83
//消息摘要
SHA1(M) =
        42 76 5E B4 FC 0D 9E A1 75 76 7A A5 1F D9 98 35
        1C 02 B8 6C
//RSA 参数 N
N =
        CB 84 92 0E FA 95 92 66 6F 5B 87 DD 21 92 3C 3E
        24 B7 87 2B A5 41 D5 78 00 0F A0 72 AA 7A 29 11
        75 35 AC 62 4A 28 A7 1D AB 2B 10 AD 58 D4 C0 86
        6D E5 B2 07 1D CA BC E7 86 75 A4 8E 35 D0 24 4F
        61 C0 3C 66 0F 85 A4 0D A7 9A 19 A1 15 5D 8E CD
        DB 34 37 3F 68 1D BE B0 B3 F1 75 59 22 04 DF BA
        99 A8 EE 55 B0 D0 99 6F F5 50 D6 31 8F 05 7C B3
        9F 34 5A 01 44 02 2D 6E C8 51 B4 DA 7E DE DD C9
//RSA 参数 E
E =
        01 00 01
//RSA 参数 D
D =
        65 31 33 96 73 0E 2F CE F3 0B A5 D0 53 C2 EC 65
        51 C8 57 53 62 46 0B A8 31 0F 94 0D AF AA 32 05
        56 2C 2B DC 6B 57 2F 50 D7 2D 00 8B D0 A1 68 60
        EE FF C8 B7 35 1E FB 32 7B 6E BF 35 5E 98 71 94
        8B DC 79 B4 28 83 DC DA 30 03 FB EB 46 F7 32 E1
        13 B4 8E 4F 69 C2 51 F2 0E 71 8E AB 75 93 94 44
        FB 2F 07 B4 6D 5B 51 21 EE FD A5 1A 3A 7E BC C8
        D3 E8 8F E2 1B 57 96 13 B0 6E 7F 50 B5 02 4E 29
//签名值
S = M^D mod N =
        7E 98 27 5B 91 3A 7A 3C 1D 8F D7 73 84 C3 E7 59
        83 A2 C1 80 EE 65 96 15 CD 6B BC 03 4D 80 87 4B
        48 C9 8A D7 EB 7E 8F D0 21 E1 9D 22 A7 DD 3E BB
        30 A4 AA B0 6E F4 D5 17 BD 6B 1E D1 1A F3 31 74
        B5 7D 85 45 2E 5A 10 29 EA 57 B1 3C 0C 4D 9B 56
        86 0F 9C 97 17 06 04 2B A8 36 F1 D3 B6 52 FC 89
        88 2A 28 C3 30 CC 03 74 CD 44 13 72 E3 30 AB FD
        1B B6 D3 15 E4 3B E4 94 CD 92 39 72 20 22 90 11
```

2. ECDSA 签名举例

```
消息的哈希值 S(M) =
        42 76 5E B4 FC 0D 9E A1 75 76 7A A5 1F D9 98 35
        1C 02 B8 6C
私钥 d = 03
公钥 Q =
        90 51 DA E6 86 FA 68 10 3A 47 8D B3 98 81
        8D 04 8C 20 42 F0 1F 0E CA B5 77 E4 59 8E
```

```
      BA A9 7F 6F 99 CA BF 4A 62 6C 05 6B 63 F2
      1F 79 35 89 D3 6C D5 98 1A 35 79 78 29 00
随机数 r = 12 34
(r,s) =
      3A DD FF B7 A5 C9 00 84 78 82 8A 30 70 76
      FD 62 4A 83 D2 01 9E 09 5C 7D C4 BE A7 2A
      2E D8 B3 24 DE 29 9D 9A C6 89 1F 05 FA 9D
      E1 CB 6C 69 80 1C 73 36 53 6F 08 FE CE E3
验证过程:
v =
      3A DD FF B7 A5 C9 00 84 78 82 8A 30 70 76
      FD 62 4A 83 D2 01 9E 09 5C 7D C4 BE A7 2A
r =
      3A DD FF B7 A5 C9 00 84 78 82 8A 30 70 76
      FD 62 4A 83 D2 01 9E 09 5C 7D C4 BE A7 2A
v = r,验证成功!
```

参考文献

[1] 刘建伟,王育民.网络安全——技术与实践.北京:清华大学出版社,2005

[2] Behrouz A. Forouzan 著,马振晗,贾军保译.密码学与网络安全.北京:清华大学出版社,2009

[3] International Standard ISO/IEC 7816-9. Identification cards. Integrated circuit cards, Part 9: Commands for card management,2004

[4] International Standard ISO/IEC 7816-4. Identification cards. Integrated circuit cards, Part 4: Organization,security and commands for interchange,2005

[5] ICAO DOC9303. Part1 Machine Readable Passports. Volume 2 Specifications for Electronically Enabled Passports with Biometric Identification Capability. Sixth Edition. International Civil Aviation Organization. 2006

[6] Bruce Schneier.应用密码学.北京:机械工业出版社,2000

[7] IETF Internet Public Key Infrastructure X.509 Certificate and CRL Profile. ftp://ftp.isi.edu,1988

[8] 丁士明,刘连忠,陆震.一种基于 USB-KEY 的身份认证协议.微机发展,2005,15(10):1～3

[9] 潘娟.基于 PKI 的身份认证技术的研究和实现.北京:华北电力大学,2002

[10] 岳佩.智能卡数据交互安全性的研究与实现.北京:北京交通大学,2008

[11] 亨德利著,杨义先等译.智能卡安全与应用.北京:人民邮电出版社,2002

[12] Mike Hendry.智能卡安全与应用.第 2 版.北京:人民邮电出版社,2002

[13] 范晓红,吴今培,张其善.智能卡文件系统的安全访问机制.微计算机应用,2004,25(1):37～42

[14] 刘守义.智能卡技术.西安:西安电子科技大学出版社,2004

[15] 杨义先,钮心忻.应用密码学.北京:北京邮电大学出版社,2005

[16] 刘嵩岩,毛志刚,叶以正.智能卡的研究与发展.微处理机,2000,(2):1～5

[17] 王爱英.智能卡技术.北京:清华大学出版社,2000

[18] 亓文华.终端可信数字签名关键技术的研究及实现.北京:北京航空航天大学,2007

[19] 胥怡心,张其善.Java 卡对象共享安全策略分析与实现.计算机应用,2009,29(6):1615～1621

[20] 刘玉珍,涂航,张焕国等.实用智能卡操作系统的设计与实现.武汉大学学报,2000,46(3):309～312

[21] 吕英豪,刘飞飞.多应用智能卡系统安全及应用研究.南方冶金学院学报,2003,24(1):70～74

[22] 曹化工,梁宗炼等.基于智能卡的 PKI 体系实现框架.小型微型计算机系统,2003,24(6):1005～1008

[23] Biham E, Shamir A. Differential Fault Analysis of Secret Key Cryptosystems. In: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology table of contents, Springer-Verlag, 1997, 1294: 513～525
[24] Schindler W. A Combined Timing and Power Attack. Public Key Cryptography, 2002, 2274: 263～279
[25] Ludger Hemme. A Differential Fault Attack against Early Rounds of (Triple-) DES. Cryptographic hardware and embedded system, 2004, 3156: 254～267
[26] 朱新山,高勇,王阳生. 机器可读旅行文档的安全分析. 计算机科学,2005,32(11): 130～133
[27] 荣彦,贺惠萍. 基于 PKI 规范的通用认证系统的设计与实现. 微计算机信息,2006,22(5-3): 55～57
[28] 冯清枝,王志群. 智能卡的安全机制及其防范策略. 中国人民公安大学学报,2004,(1): 95～97
[29] 袁晓宇,张其善. 基于智能卡的 RSA 数字签名实现关键问题解析. 电子学报,2004,2(11): 1897～1900
[30] ISO/IEC 9594-8/ITU-T Recommendation X. 509. Information technology. Open Systems Interconnection. The Directory: Authentication framework, 1997
[31] Housley R, Ford W, Polk W. RFC 2459: Internet X. 509 Public Key Infrastructure Certificate and CRL Profile, 1999
[32] 李胜广,张小波,张之津等. 深度剖析电子护照扩展访问控制安全机制. 2009 年公安部第一研究所论文论文集,2009
[33] 李胜广,薛艺泽,张之津. 机读旅行证件攻击分析与安全策略研究. 警察技术,2009,(2): 37～40



生物特征识别

智能卡是个人身份信息和财产信息的数字化载体。载体的安全取决于智能卡安全机制的应用,但有时仅依靠安全算法和密钥认证还不足以提供充分的个人身份鉴别。生物识别技术成为了弥补密钥认证机制的最佳方法,在智能卡领域中得到了广泛的应用。生物识别技术是根据每个人自身具有的生物特征来进行身份验证和识别的一种方法。

生物特征识别有时候也称生物识别或生物认证,都是指通过获取和分析人体的身体或行为特征来实现身份的自动鉴别。从生理特征的角度可分为指纹识别技术、人脸识别技术、虹膜识别技术、掌形识别技术、人耳识别技术、视网膜识别技术等等(生物识别技术间的比较参见表 6-1)。以上这些生物特征具有人各有异、终身不变和随身携带的特点。

表 6-1　多种生物识别技术特点对比表

类型	鉴别可靠度	可否运用一对一	可否运用一对多	传感器价格(人民币,元)	传感器尺寸
指纹	很好	是	是	100～1000	非常小
虹膜	很好	是	是	1000～10 000	大
面部	好	是	是	1000	小
掌形	较好	是	否	1000	中等
人耳	较好	是	否	1000～10 000	小
签名	一般	是	否	1000	小

近几年来,生物识别技术已从研究阶段转向应用阶段,对该技术的研究和应用进行得如火如荼,前景十分广阔,尤其是在身份识别和安全应用领域,该技术被广泛应用,成为一种越来越受欢迎的安全保障措施,随着社会对身份识别要求的不断提高,该技术的重要性将日益凸现。

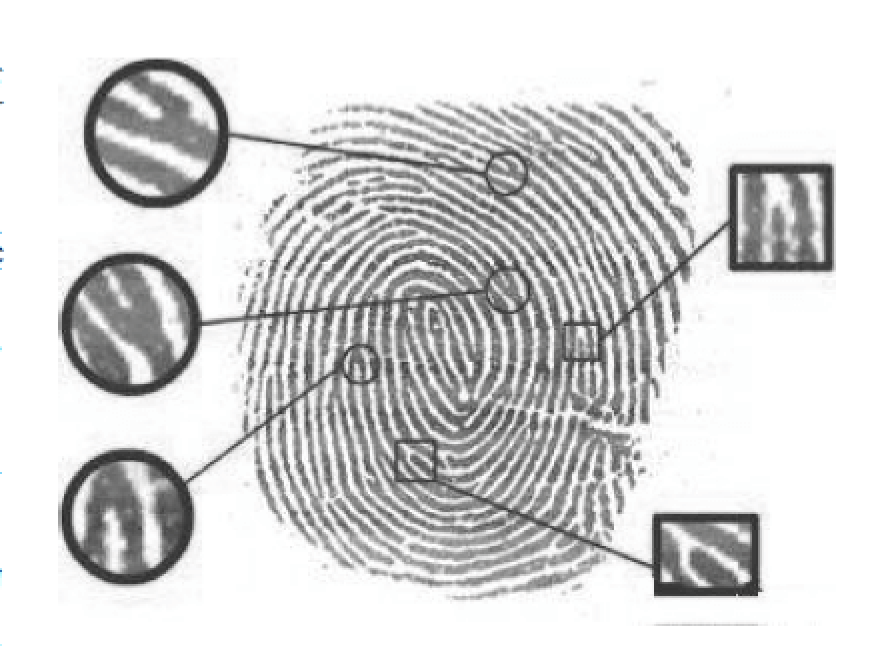
6.1　指纹识别技术

6.1.1　概述

指纹学的研究开始于 16 世纪末。20 世纪初,指纹识别技术被正式作为一种身份认证的方法。指纹识别指通过比较不同指纹的细节特征点来进行鉴别,是目前最为成熟的、应用最为广泛的生物识别技术。

指纹是手指皮肤表面隆起的脊线和凹下的谷线构成的特定纹路,其纹理在婴儿胚胎时期就已经确定,终身不变。指纹由皮肤表面死亡的角质细胞堆积而成,即使磨损,仍能重新

长出。包括指纹在内的皮肤纹路在图案、断点和交叉点上各不相同，图 6-1 为指纹特征点示意图。


图 6-1　指纹特征点示意图

这些都为指纹识别技术奠定了重要的物理基础。指纹识别技术具有识别准确率高、安全、方便、应用场景广泛、设备成本低廉等诸多优点。

指纹识别技术通过图像采集设备读取指纹图像，然后用计算机识别和分析指纹的全局特征和局部特征，如脊、谷、终点、分叉点和分歧点等，再从中抽取特征值，从而非常可靠地通过指纹来确认一个人的身份。具体过程如图 6-2 所示。

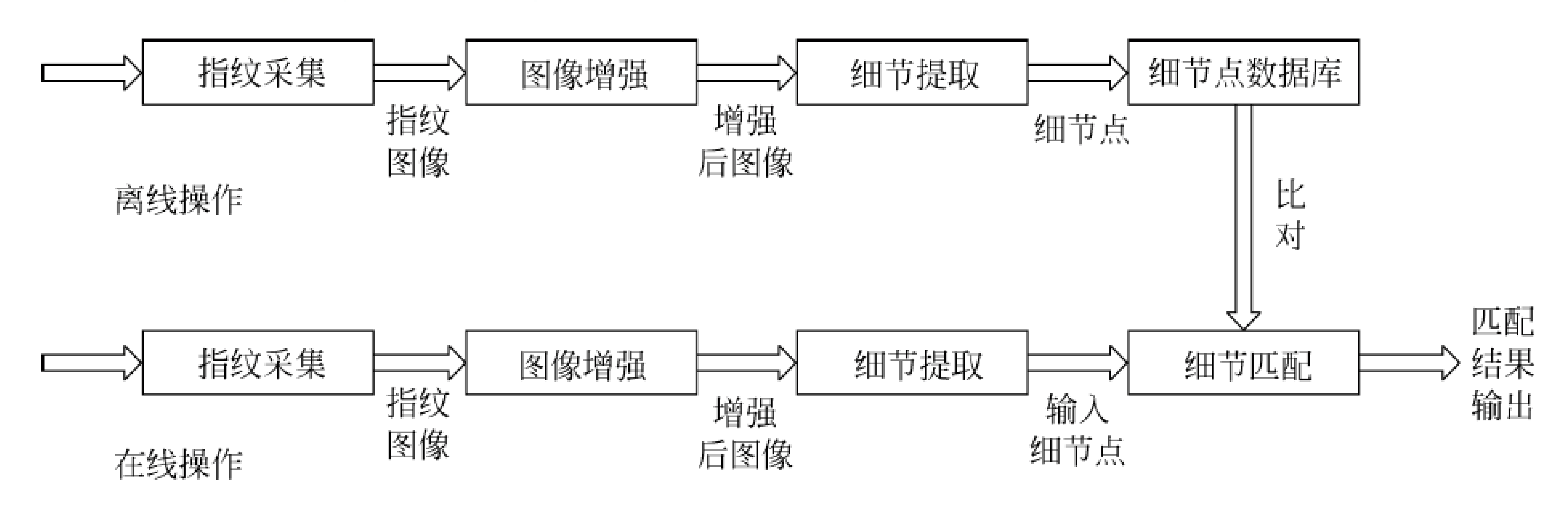


图 6-2　指纹识别过程图

下面将对图 6-2 所示的识别过程和关键技术进行介绍。指纹识别中的关键技术包括指纹图像的采集、预处理、特征提取和特征匹配。

6.1.2　图像采集

随着光学仪器、传感器及数字技术的发展，各种快速、精确、方便、小巧的采集设备得到了广泛的应用。常用的指纹图像采集装置有如下内容。

(1) 基于光学全反射技术的指纹图像采集器(如图 6-3(a)所示)。该采集器利用激光照在手指上，然后用 CCD 阵列获取其反射光，由于反射光随着指纹的脊线和谷线的深度不同而不同，因此可以得到指纹图像。



(a) 基于光学全反射技术


(b) 基于超声波扫描技术


(c) 基于固态阵列传感器技术

图 6-3　三种常见的指纹图像采集设备

(2) 基于超声波扫描技术的指纹图像采集器(如图 6-3(b)所示)。该采集器利用指纹脊线和谷线的深度不同导致超声波的反射回波不同的原理来产生指纹图像。

(3) 基于固态阵列传感器技术的指纹图像采集器(如图 6-3(c)所示)。该类采集器利用大量的敏感元件组成固态阵列芯片,比如电容传感器、热敏传感等。通过感受按压指纹的压力、热度等特征来摄取指纹。

6.1.3 图像预处理

指纹识别需要提取指纹中的有效特征,而特征提取的性能很大程度上要依赖于指纹图像的质量。然而,在实际应用中,由于采集条件(指纹太湿、太干或比较脏)和采集设备的因素,采集到的指纹图像质量比较差,含有大量的噪声,影响后续处理的效果。因此,在对图像进行特征提取前,必须经过一系列的预处理,消去大量噪声信号,以便得到纹线清晰的点线图,为后续工作奠定基础。

典型的指纹图像预处理包括图像规格化、图像分割、图像增强、二值化去噪和图像细化等,整个指纹图像预处理的流程如图 6-4 所示,其最终结果是得到一幅清晰的点线指纹图。

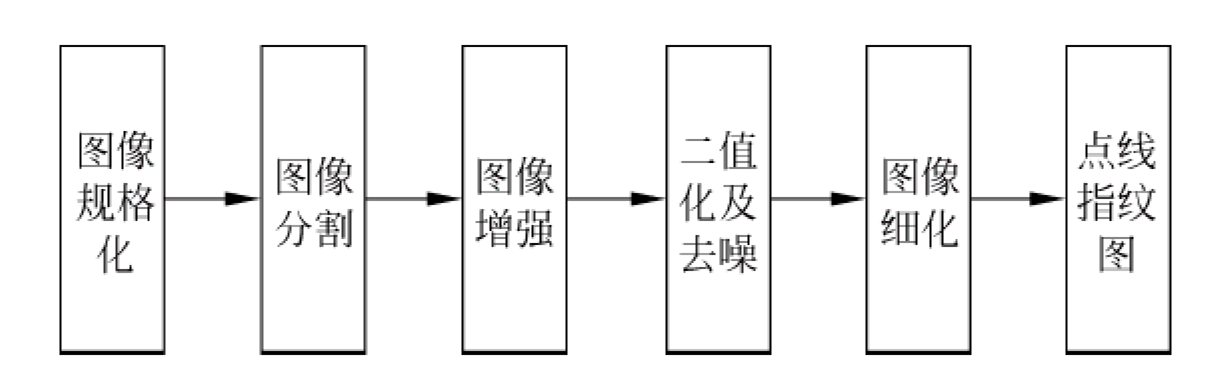


图 6-4　指纹图像预处理流程图

图像预处理过程各阶段详细描述如下。

1. 指纹图像规格化

指纹图像规格化的目的是消除传感器本身的噪声以及由于手指压力不同而造成的灰度差异。该过程可以将不同的源图像的对比度和灰度调整到一个固定的灰度级别上,为后续图像处理提供一个较为统一的图像规格,对于大小为 $M\times N$ 的灰度图像,基本步骤如下。

(1) 计算整幅图像的均值和方差

$$M(Q)=\frac{1}{M\times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}Q(i,j)$$

$$\operatorname{var}(Q)=\frac{1}{M\times N}\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}(Q(i,j)-M(Q))^2$$

$Q(i,j)$为像素点(i,j)的灰度值。

(2) 按如下公式对灰度图像进行规格化

$$G(i,j)=\begin{cases}M_0+\sqrt{\dfrac{\operatorname{var}_0(Q(i,j)-M)^2}{\operatorname{var}(Q)}} & Q(i,j)>M\\ M_0-\sqrt{\dfrac{\operatorname{var}_0(Q(i,j)-M)^2}{\operatorname{var}(Q)}} & \text{其他}\end{cases}$$

其中,M_0,$\operatorname{var}(Q)$分别为期望的均值和方差。

2. 指纹图像分割

图像分割是预处理中的一个重要步骤,它将指纹的有效区域从背景和噪声中分离出来。

有效区域的分割不仅简化了后续的处理，更显著提高了细节特征提取的可靠性。

分割算法可分为如下几类：

(1) 基于方差阈值的分割方法。相对于其他区域，有效的指纹区域具有一个较高的方差值。

(2) 基于方向场信息的分割方法。它的性能依赖于方向场的可靠性。灰度的对比度不敏感，实际上在脊线不连续的区域、中心点和奇异点附近的区域中，精确地提取方向场几乎是不可能的。

(3) 基于频域的分割方法。对于指纹弹性形变导致脊线间隔不均的区域，这种算法尚不能很好地处理。

综上所述，虽然基于方差阈值的方法被广泛采用，但单一的采用以上分割方法并不能得到理想的处理效果，为了解决在分割处理中存在的不确定因素，人们也提出了一系列的改进算法，比如基于 D-S 证据理论的分割、用 Urn 模型实现图像分割，Markov 模型的分割等。

3. 指纹图像滤波增强

指纹图像的增强主要是对指纹灰度图的滤波增强处理，其主要作用是在尽量保持图像中纹线边缘完好的前提下，去除指纹图中的叉连、断点及模糊不清的部分，以减少指纹图像在识别过程中可能造成的计算与分析误差。常用的图像滤波增强方法有均值滤波、中值滤波、对比度调整法和方向图滤波等。

4. 图像二值化

二值化处理是将一幅灰度图像转化为二值图像。它提高了指纹图像中脊线和谷线间的对比度，使细节点的提取更加方便。二值化的关键问题是选取一个合适的阈值。二值化方法主要有固定阈值法和局部自适应阈值法，但由于指纹图像各个部分的明暗程度不同，所以对一幅指纹图像采用一个固定的二值化阈值得不到很好的效果，因此普遍采用基于局部适应阈值的算法来生成二值化图像。

下面简单介绍两种常用的二值化方法。

(1) 基于方向场的局部阈值二值化方法

若该像素处的脊线方向为 i，先计算该像素处在方向 i 和垂直方向 $\text{iVar}=(i+4)\bmod 8$ 的灰度平均值 Gmean[i]和 Gmean[iVar]；然后将该像素二值化为

$$\text{iVar} = \begin{cases} 255 \text{Gmean}[i] \geqslant \text{Gmean}[\text{iVar}] \\ 0 \quad \text{其他} \end{cases}$$

其中 iVar 表示二值图像中该像素处的值，255 为位置图像中图像背景和谷线。

(2) 基于低通滤波的局部阈值二值化方法

假设指纹增强后的图像上任意一点的像素值为 $G(i,j)$，用于公式对图像进行低通滤波

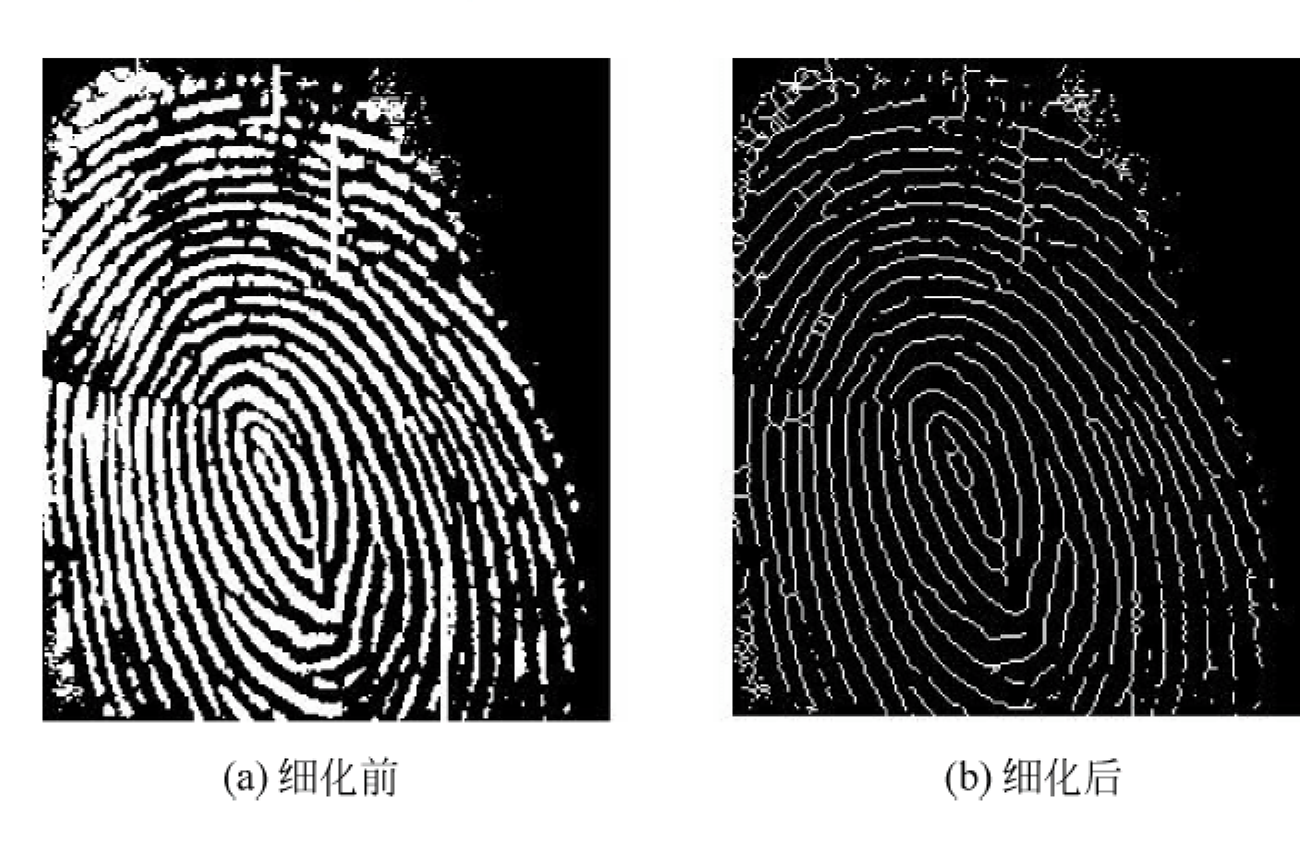
$$G'(i,j) = \sum_{u=i-\frac{W}{2}}^{u=i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{v=j+\frac{W}{2}} G(u,v)$$

G'就是滤波后的图像。

5. 指纹图像细化

指纹细化是把二值化后清晰但纹线粗细不均匀的二值指纹图像转化为纹线宽度仅为一个像素的条纹，如图 6-5 所示，两幅图分别为细化前后的效果。细化后的指纹图像保证了纹

线的简洁性和方向性，而且特征点不变，中心线基本不变。细化算法的种类很多，根据算法的迭代方式不同，可分为串行算法和并行算法。在串行细化算法中，每次迭代的结果不仅取决于前一次的迭代结果，而且与当前处理情况有关；而在并行方式中，当前迭代则仅仅由上次的迭代情况决定。因此，并行方式优于串行方式。



(a) 细化前　　(b) 细化后

图 6-5　细化前后效果对比图

6.1.4　特征提取

指纹图像识别的一个重要过程是指纹的细节特征提取，它是对预处理完成所得到的二值细化指纹图像进行细节特征点的提取过程。在人类的指纹中存在两类不同的特征，即全局特征和局部特征。全局特征是指人眼可以直接观察到的特征，这类特征常用于指纹的分类，因此也被称为分类特征，主要包括中心点(core)、三角点(delta)等。而局部特征则指指纹上的细节点，常用于指纹的识别，主要包括端点(endpoint)、分叉点(bifurcation)、孤立点(dot)、环点(loop)、短线(short ridge)等。因此提取的特征点的好坏，直接影响了指纹的识别效果。指纹特征提取就是要从指纹图像中提取出有代表性的特征。

1. 细节提取

局部特征也被称为细节特征。虽然任意的两枚指纹可能有相同的全局特征，但它们的细节特征不可能完全相同，所以细节特征的提取在指纹识别中至关重要。目前，指纹细节特征点提取的方法主要有 3 种：

(1) 直接从灰度图像中提取细节特征点。

(2) 从二值化图像中提取细节特征点。

(3) 从细化后的图像中提取细节特征点。

前两种方法预处理步骤较少，但是特征提取算法复杂，而且由于噪声等因素的影响，特征定位也不够准确。因而大多数系统采用从细化二值图像中提取细节特征的方法。对于细化后的指纹二值图像，每个像素点的灰度值只有 0、1 两种情况(0 表示背景点的灰度，1 表示纹线点的灰度)。通过观察细化后的指纹图像发现，细节特征点的类型与周围八个方向点的像素值之间有一定的规律，在这里采用如图 6-6 所示的 3×3 的模板扫描细化后的指纹图像，通过公式计算该点的八邻域点灰度值由 0 变为 1 或由 1 变为 0 的次数(用

X_8	X_1	X_2
X_7	P	X_3
X_6	X_5	X_4

图 6-6　八邻域图

C_P 表示)，可以得到该点的类型并记录它们的位置。

$$C_P = \frac{1}{2}\sum_{k=1}^{8} | R(k+1) - R(k) |, \quad R(9) = R(1)$$

其中 $R(1), R(2), \cdots, R(8)$ 是细化后像素 P 点沿顺时针方向排列在 $X_1, X_2, \cdots, X_8$ 处的灰度值。如 $C_P=1$，则点 P 为脊线端点，如 $C_P=3$，则点 P 为脊线分叉点。

2. 后处理

在指纹识别系统中，指纹端点和分叉点是后续指纹匹配时最重要的细节特征点。但事实上，在指纹输入时，由于受汗渍、干燥、按压轻重的不同等影响，得到的图像通常含有断纹、细化等预处理算法易引入的噪声，因而提取的细节特征点中往往包含了大量的伪特征点。伪特征的数据达到一定的程度，将会严重影响后续操作，使匹配算法复杂度增加，甚至会影响整个识别系统的正确率。

常见的伪特征有毛刺、短线、断线、小桥、小孔等，如图 6-7 所示。所谓后处理就是尽可能滤除伪特征、保留真特征而进行的操作，这对整个自动指纹识别系统来说是十分必要的。后处理操作分为两种情况：一种是在特征提取前，对预处理后的指纹图像进行去除毛刺、连接断纹等操作，然后提取特征作为真特征；另一种是在特征提取之后，根据特征之间的相互关系，尽可能准确地识别伪特征点并滤除它们。前者直接对图像进行修补，操作比较复杂，容易引入新的伪特征；后者对特征提取后的数据进行判断，识别比较麻烦，但是速度较快。

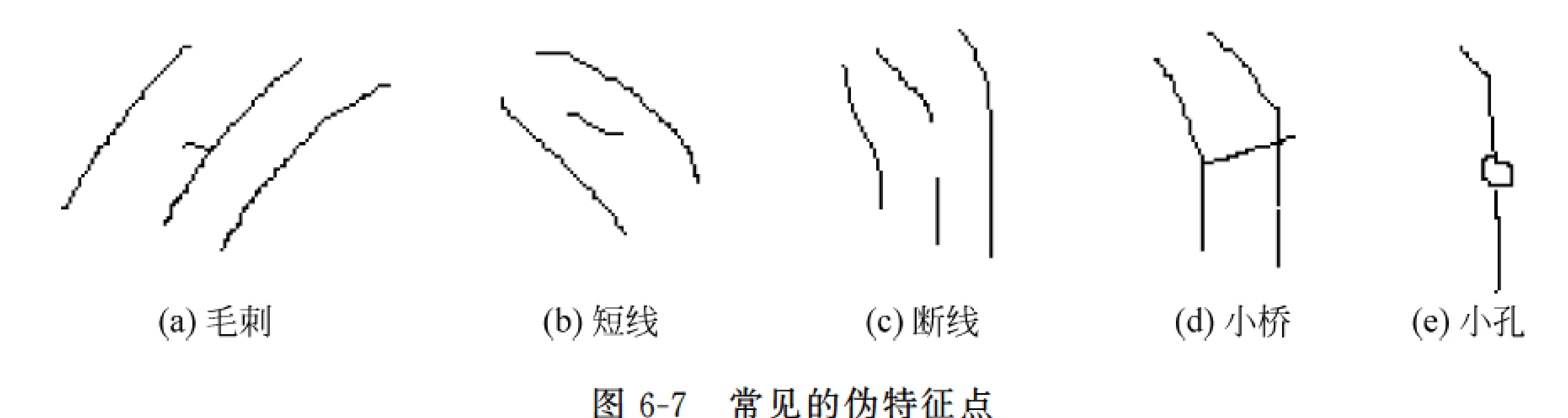


图 6-7 常见的伪特征点

6.1.5 指纹分类

在一次有效的指纹识别过程中，需要将被识别指纹与样本数据库中的大量指纹进行比对，当样本数据库的容量非常大时，这种比对将会非常耗时。通常根据指纹的整体特征将指纹分成几类，来减少搜索的时间，降低计算的复杂程度。从某种意义上讲，指纹分类过程是一次粗糙的指纹匹配过程，实际上是为指纹识别提供了一个索引机制。目前常用的指纹分类方法有基于统计法的指纹分类方法和基于结构法的指纹分类方法。

(1) 统计法是指直接从输入图像的方向信息中获得特征向量对指纹进行分类。

(2) 结构法是指利用一些突出的指纹特性和它们之间的关系对指纹进行分类。

但是由于指纹纹线形态的复杂性，指纹专家对作为分类标准的全局特征的定义是非常复杂和模糊的，要想使用这些复杂和模糊的分类标准对指纹准确地进行自动分类是非常困难的，所以迄今为止，自动指纹分类技术还停留在理论研究阶段，离实际应用尚有一定距离。

6.1.6 特征匹配

在指纹识别系统中，指纹匹配是完成识别最关键的一步，也是评价指纹识别系统性能好

坏最主要的依据。指纹匹配是根据提取的指纹特征，来判断两枚指纹是否来自于同一个手指。特征比对精确度的高低和速度的快慢，对整个指纹识别系统的影响很大。为了能够实现准确、快速地识别指纹，以保证自动指纹识别系统的实时性，指纹匹配算法至关重要。现在常用的匹配算法包括基于点模式的匹配方法、基于纹理模式的匹配方法、基于混合特征的匹配方法和基于图的匹配方法。下面将对部分常用方法进行简单的介绍。

1. 点模式匹配方法

点模式匹配是基于相似度的经典指纹细节点匹配算法。一般利用几何关系判断两组细节点集位置特性的相似程度，采用打分的方法输出匹配结果，即相似度越高得分越高。然后根据实际应用的需要设定阈值：匹配分数大于阈值，认为两幅指纹匹配成功；反之，匹配失败。具体步骤如下：

(1) 首先通过迭代过程，寻找细节点集的最佳匹配点对，使得根据该点对得到的相对变换可达到较大的匹配相似度。

(2) 然后根据相同变换下两个点集中其余点的匹配程度，计算匹配度。变换方法不同，匹配的过程也不相同。

一般认为，点模式匹配(细节特征匹配)采用很小的特征模板，却有较高的鲁棒性和较快的匹配算法，是比较合理的方法。但对于数量不等的离散点集，点模式匹配问题仍旧是模式识别的研究问题之一。

2. 纹理匹配方法

基于纹理模式的匹配能够克服基于细节点方法的一些缺点，作为一种新的匹配思路正在受到关注和应用。1999 年 C. I. Watson 等人将耐形变滤波器用于弹性扭曲指纹的匹配，获得较好的效果。2001 年 Precise Biometrics 公司采用 PPM(precise pattern matching)匹配算法将细节点临近区域的纹理结构和低曲率半径用于匹配，算法稳定可靠。

2000 年 A. K. Jain 等提出一种基于 Gabor 滤波的纹理匹配算法。该算法首先确定指纹图像中感兴趣的区域并将这块区域网格化，然后用 Gabor 滤波沿八个不同的方向处理图像，获取指纹的整体和局部信息，并得到一个固定长度的代码(finger code)，最后比较两幅待匹配指纹图像相应代码欧式距离的差异，从而确定匹配程度。

纹理匹配方法充分地利用了丰富的脊线信息，在一定程度上可以克服质量较差的区域细节点难以提取的困难，在某些应用领域可以弥补细节点匹配的缺陷。但由于这种方法需要对图像做多次卷积，运算量很大，且难以处理较大形变指纹图像的匹配。

3. 混合匹配方法

这里简单介绍一下由张堂辉提出的一种基于混合特征的匹配算法，分析各种特征之间的互补性。由于细节点算法在图像校准方面比较准确，而指纹中心点提取算法的可靠性比较差，所以在进行纹理匹配前采用细节点算法的校准输出结果，计算出两幅图像的重叠区域。之后引入一个反馈匹配环节，将原来不在重叠区域的局部细节特征去除，然后重复细节点算法的匹配过程，匹配中把奇异点作为细节点来处理，赋予奇异点较大的权重值，最后计算出细节匹配的指纹相似度 S_m。如果 S_m 达到某个阈值，那么将继续后面的纹理匹配过程，如果细节匹配的结果已经非常明显的相似或者不相似，决策结果就不需要再参照纹理匹配的结果，这样处理可以提高匹配速度。

6.2 人脸识别技术

如同人的指纹一样，人脸也具有唯一性，可用来鉴别一个人的身份，人脸识别技术在商业、法律和其他领域有着广泛的应用。人脸识别首先是法律部门打击犯罪分子的有力工具，在毒品跟踪、反恐怖活动等监控中发挥着重要的作用。人脸识别的商业应用价值也正在日益增长，主要是信用卡或自动取款机的个人身份核对，与利用指纹、手掌、视网膜、虹膜等其他人体生物特征进行个人身份鉴别的方法相比，人脸识别具有直接、友好、方便的特点，特别是对于个人来说没有任何心理障碍。

人脸识别技术是指通过面部特征来进行身份识别的一种生物特征识别技术。人脸识别技术是一个跨学科技术，它涉及图像处理技术、视频处理技术、神经网络理论以及统计学习理论等。该技术主要包括人脸检测和人脸识别两个主要过程。人脸检测是指在输入图像中确定所有人脸的位置、大小、位姿的过程。人脸识别是根据已经检测出来的人脸来进行身份判别，它主要包括人脸辨别和人脸确认两大部分。

人脸识别系统的框图参见图 6-8。

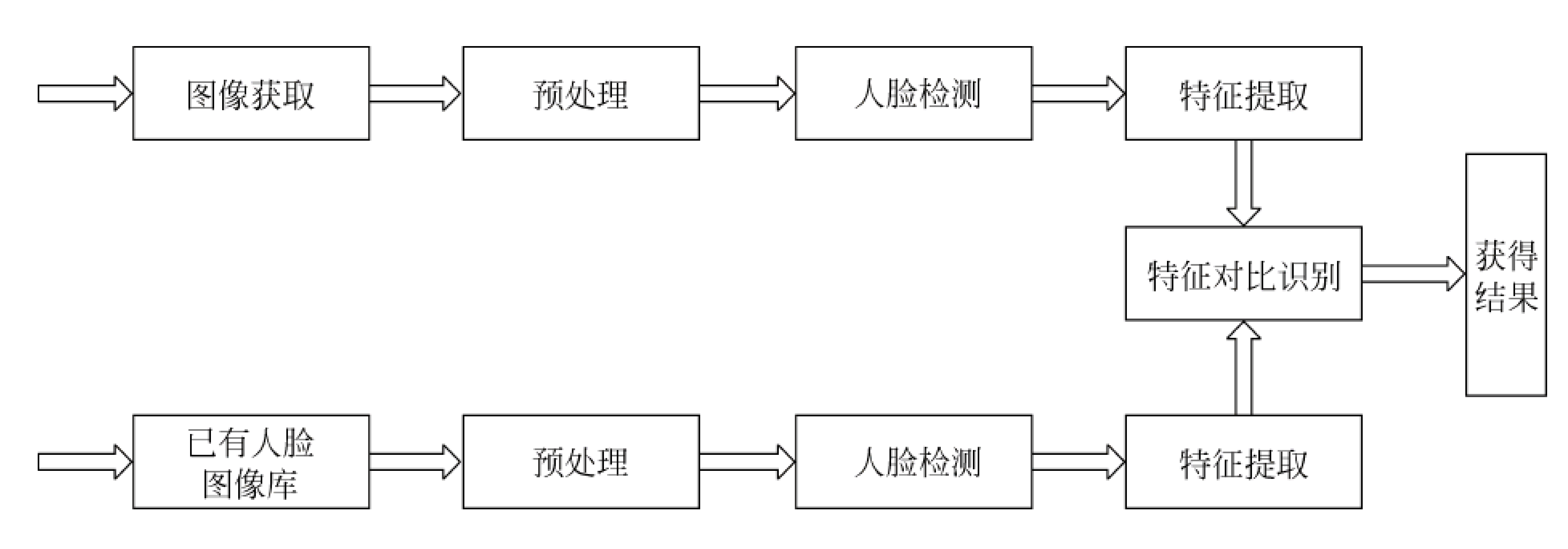


图 6-8 自动人脸识别系统框图

在 20 世纪 60 年代初期，人脸识别就引起了研究者的强烈兴趣。最近几年来，人脸识别研究越来越受到学术和商业界的关注，人脸识别的输入图像通常有三类情况：正面、侧面、倾斜。国内关于人脸自动识别的研究始于 20 世纪 80 年代，我国许多高校、研究机构在图像处理和模式识别领域有很好的研究基础，积极开展了对包括人脸识别在内的基于人体生物特征识别技术的基础研究和应用开发工作。人脸识别技术有着非常广阔的应用前景，自动的人脸识别系统在各种不同领域中的应用必将对人们的生活产生深远的影响。

6.2.1 人脸检测技术概述

人脸检测（face detection）的目的是在给定的任意图像或图像序列中，确定图像中是否存在人脸。如果有，定位出人脸的位置和空间范围。人脸检测的三个基本目标是：确定是否存在人脸、人脸的数目、人脸的位置和尺寸。

人脸检测问题所包含的内容十分广泛，从不同的角度可以有多种分类方法。本节主要讨论静止图像中的人脸检测问题，对于动态情况应属人脸跟踪范畴。同样，人脸检测的方法也有多种分类。归纳起来，根据利用特征的色彩属性可以将人脸检测方法分为基于肤色特

征的方法和基于灰度特征的方法；根据人脸检测时采用的不同模型，又可以将基于灰度特征的方法分为基于启发式(知识)模型的方法和基于统计模型的方法。前者主要利用人们的先验信息构造人脸检测器，后者主要利用机器学习和概率统计的方法构造人脸检测器。由于人脸检测问题的复杂性，无论哪一类方法都无法适应所有的情况，一般都是针对人脸检测领域内某个或某些特定问题采用某一特定方法。

目前国外对人脸检测研究较多，比如 MIT(麻省理工学院)，CMU(卡内基·梅隆大学)等。国内的公安部第一研究所、清华大学、中科院计算所和自动化所也在做相关领域的研究工作。

下面对各种检测方法做简单的介绍。

6.2.2 基于启发式模型的检测方法

基于启发式模型方法首先抽取几何形状、灰度、纹理等特征，然后检验它们是否符合人脸的先验知识。在启发模式下又分为以下 3 种方法。

1. 基于先验知识的方法

基于先验知识的方法是将人脸面部器官之间的关系编码准则化的人脸检测方法。该方法是一种自上而下的方法，依据人脸面部器官的对称性、灰度差异等先验知识，制定出一系列的准则。当图像中的待测区域符合准则，则被检测为人脸。

Sakai、Yang 等在 1994 年提出的基于镶嵌图的人脸检测方法就是基于先验知识方法的典型例子。该方法利用 4×5 镶嵌图将人脸分块，根据每块的灰度值制定准则来进行判定。他们将系统分为三级，利用不同精度的二次采样产生三级不同分辨率的图像(如图 6-9)。针对不同分辨率的图像采用不同的准则进行判定，例如：在低分辨率图像里的准则主要体现了人脸的大体轮廓，而在高分辨率图像里的准则主要体现了人脸的细节特征(图 6-10 为检测逻辑框图)。虽然 Yang 的方法在检测性能方面不突出，但是这种由粗至细的检测思想对以后的研究工作产生了积极的影响。

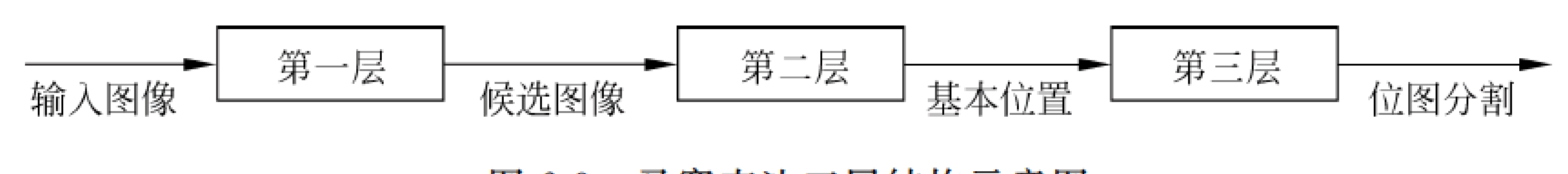


图 6-9 马赛克法三层结构示意图

卢春雨等对镶嵌图方法进行了改进，提出了 3×3 的广义三分图方法。该方法充分利用了人脸器官的自然分布，可以更直观地利用人脸的先验知识制定准则，使镶嵌图对脸形的自适应操作成为可能，并取得了较好的实验结果。基于知识方法的一个难点是如何将人类知识转化成为有效的规则，如果规则制定得太细，那么可能有许多人脸无法通过规则的验证；如果规则制定得太宽泛，那么可能许多非人脸会被误判为人脸。此外，由于列举所有的情况是一项很困难的工作，所以很难将这种方法扩展到不同位姿下人脸的检测。

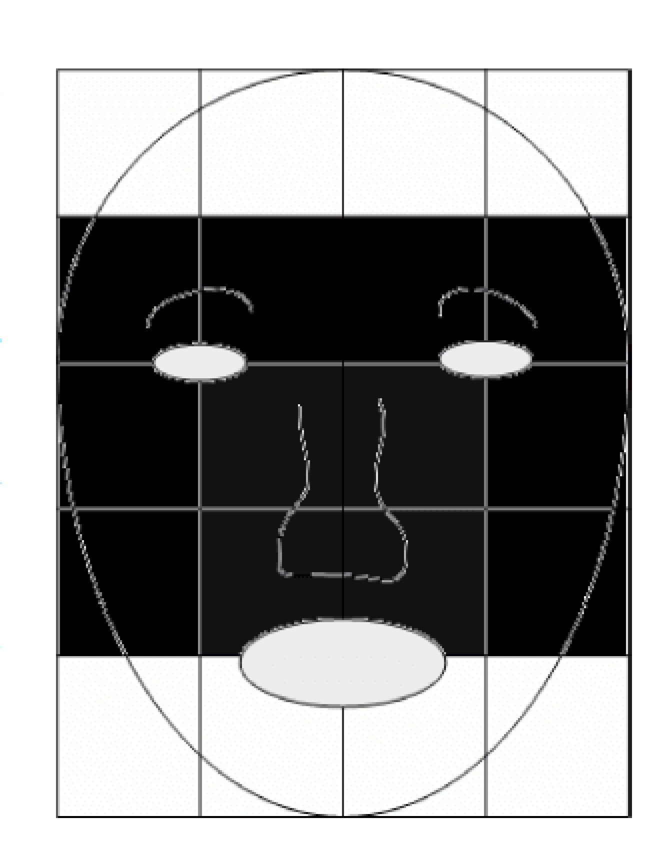


图 6-10 人脸第一层四块分图

2. 基于局部特征的方法

基于局部特征的方法着眼于检测面部的一些不变的特

征，例如眼睛、鼻子、嘴巴等。首先在整个图像中搜索一组人脸局部特征，然后通过它们之间的几何关系组合成候选的人脸区域。与基于先验知识的方法不同，该方法是自下而上的，先利用各种手段寻找上述不变特征，然后综合找到的这些不变特征来确定待检测区域是否是人脸。Yow 和 Cipolla 等对基于局部特征方法进行了改进，使它适合检测不同位置、大小和视角的人脸图像，然后根据形态学的知识设定一系列的阈值，找出眼睛、嘴巴等区域，最后依照以上位置检测出人脸。王延江等用肤色方法分割出人脸的候选区域后，然后利用小波分解对每一个候选区域进行人脸特征分析，如所检测到的区域特征分布相似于某一预先定义的人脸模型，则确认该区域代表人脸。

基于局部特征方法的主要问题是由于光照、噪声和遮挡等使图像特征被严重地破坏，人脸的特征边界被弱化，阴影可能引起很强的边缘，而这些边缘可能使算法难以使用。

3. 基于模板的人脸检测方法

基于模板匹配的方法需要预先建立一个标准的人脸模板。这个标准人脸通常是正面的，相关的参数都由人工设置或利用函数式表达。检测过程中计算输入图像和模板之间的相关系数，常用的特征包括脸的轮廓、眼睛、嘴、鼻子等，当计算出的相关系数超过了设定的阈值，则认为相应位置存在人脸。模板匹配法的关键在于人脸模板的建立。人脸模板的好坏对整个检测算法具有决定性的意义。建立模板最简单的方法就是平均法。具体处理过程如下：

（1）进行预处理，手动地预定义并参数化一个标准人脸图案。预处理要做尺度归一化和灰度归一化的工作。最简单的模板是将人脸看成椭圆。

（2）计算输入图像与标准人脸图像的相关值，这个相关值大都是独立计算脸部轮廓、眼睛、鼻子和嘴各自的匹配程度后得出的综合描述。

（3）根据相关值和预先设定的阈值来确定图像中是否有人脸。

模板匹配方法最大的优点是应用简单，运算量相对较少，但是由于人脸模板都是预先定义的，这就导致了在检测过程中无法很好地适应人脸尺寸、形状、姿态的变化。针对这个问题，研究者们提出了多分辨率模板、多尺寸模板、子模板、弹性模板等改进方法。

Yuille 等在正面人脸检测的研究中使用了子模板的方法。该方法对嘴、鼻子、眼睛和脸的轮廓分别建立子模板，并用这些子模板组成人脸模型，子模板建立在线条分割的基础上，对输入图像首先进行基于最大梯度变化的线条分割提取，然后将提取出的线条与人脸轮廓子模板进行比较以确定候选人脸的位置。最后其他子模板与候选区域的相应位置进行比较，计算相关系数并确定人脸。这种方法首先确定检测的重点区域，然后再用检测细节的子模板进行检测，这种思想被后来许多人脸检测算法采用。

张忠波提出了基于弹性形变模板（所谓形变模板，是指对眼睛、嘴巴等面部器官形状的一种参数化描述）的人脸特征提取方法。该方法使用一系列参数化的可调模板描述人脸特征，同时定义了一个能量函数来响应可调模板的参数，能量函数要根据图像的灰度信息、被测物体轮廓先验知识来设计。当使用弹性模板进行人脸检测时，根据输入图像动态调整模板的参数，并计算能量函数。当能量函数值达到最小时，由此时模板所处的位置和模板参数就可以确定人脸的具体位置。这种方法与传统模板匹配方法相比适应能力更强，但是需要在检测前根据待检测物体的特点调整参数，否则会影响收敛结果。图 6-11(a)和(b)分别为眼睛和嘴的弹性模板。

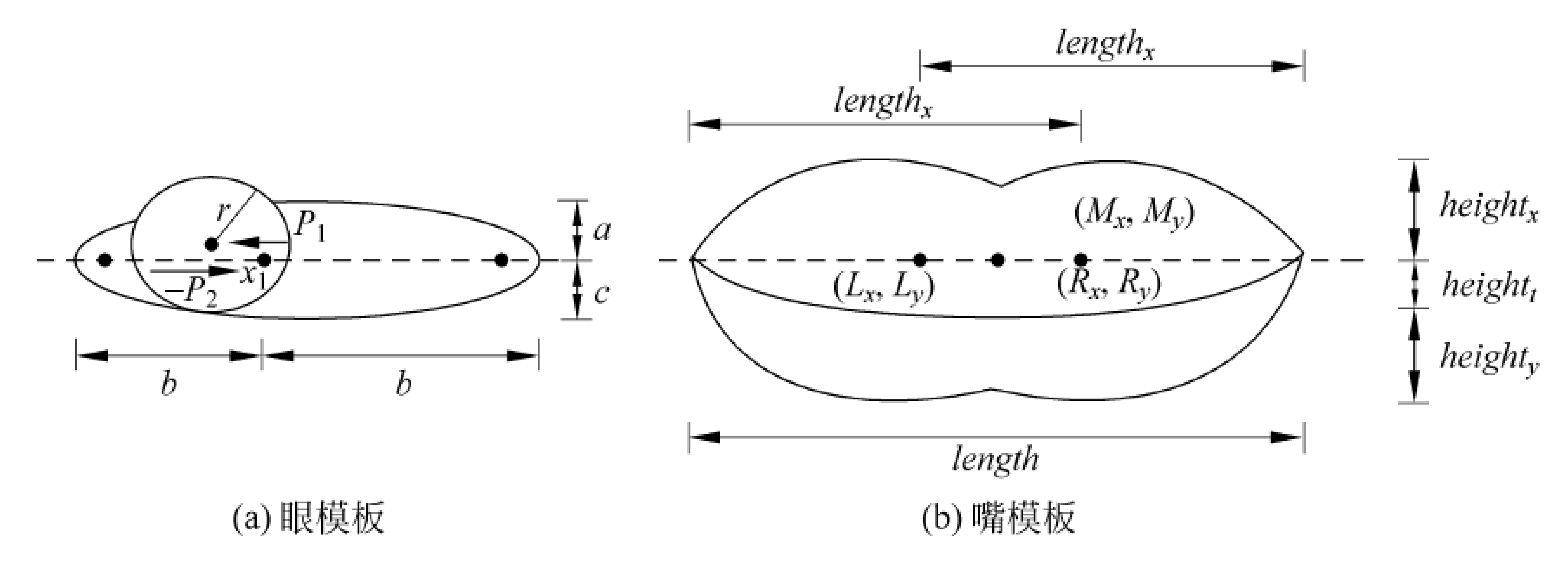


图 6-11 两种可变性模板

如图 6-11 所示，人的眼睛可以用一个圆外加两条抛物线段来表示。同时与模板的性质相对应，需要定义一个与图像中边缘(edge)、峰值(peak)、谷值(valley)等相关的能量函数。将模板动态地作用于图像，通过修改其参数使能量函数值达到最小，即通过模板的形变在图像中找到其最佳匹配。由于形变模板法利用了全局性信息，因而提高了人脸检测的可靠性，但同时也存在着很多不足之处，如初始位置难于确定、权值选取依赖经验、计算量大等。

6.2.3 基于统计模型的检测方法

基于统计模型的方法将人脸检测问题转化为模式识别的分类问题，且利用统计分析与机器学习的方法来寻找出人脸样本与非人脸样本各自的统计特征，继而构建分类器，使用分类器完成人脸检测。这种方法是目前研究的主要方向，也是将来一段时期内研究的主要趋势。其优势是不再使用人脸的特征信息等先验知识，也没有设定模板参数等操作。在统计过程中使用大量样本进行训练，使检测的结果具有一定的可靠性。从理论上讲，训练样本越多越好，所以基于统计模型的检测方法也具有一定的可扩展性。

常用的基于统计模型的检测方法有：神经网络方法、支持向量机方法、特征空间法、隐性马尔可夫模型方法、概率方法以及 AdaBoost 方法。本节主要描述前两种方法。

1. 基于神经网络的方法

人工神经网络(artificial neural network，ANN)方法是把模式的统计特性隐含在 ANN 的结果和参数中。对于人脸这类复杂的、难以显式表述的模式，基于 ANN 的方法具有独特的优势。

CMU 的 Rowley 等使用了多个 ANN 检测多姿态的人脸，算法的框架如图 6-12 所示。图中显示了两类 ANN：一个位姿检测器(pose estimator)用于估计输入窗口中人脸的位姿；三个检测器(detector)分别检测正面(frontal)、半侧面(half profile)和侧面(profile)的人脸。

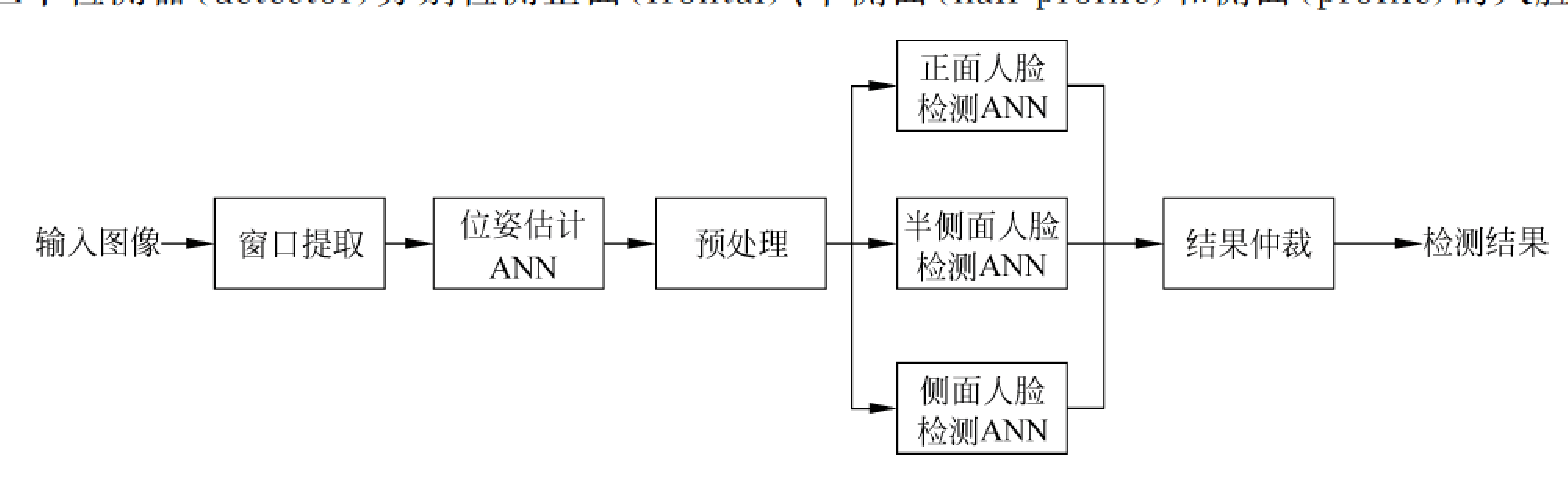


图 6-12 基于人工神经网络的人脸检测框架

使用经过对准的预处理的人脸样本以及采用“自举(bootstrap)”方法收集分类器错分的样本作为“非人脸”样本训练各个ANN,进一步修正分类器。检测时对输入图像中所有可能位置和尺度的区域首先使用位姿检测器估计人脸位姿,经校准和预处理后送入三个检测器中,最后对检测器的分类结果进行仲裁。

基于上述框架,Rowley等对正面端正人脸和正面旋转人脸的检测单独进行了研究,如图6-13所示。输入神经网络前,对每个人的脸图都进行了预处理,该方法被用于根据网络输出判断输入模式是不是人脸。

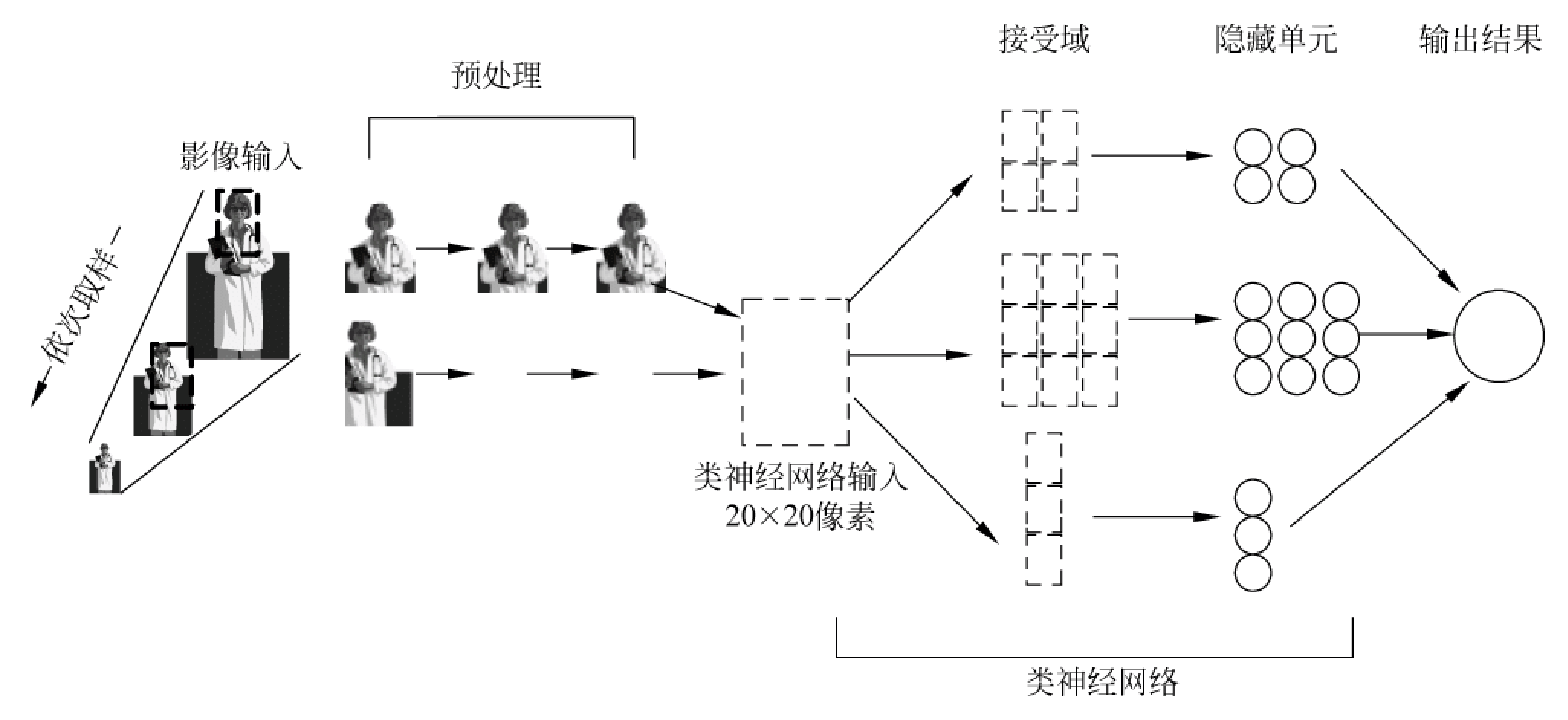


图6-13 Rowley处理系统流程图

2. 支持向量机方法

支持向量机(support vector machine,SVM)方法是在统计学习理论的基础上发展起来的一种新的模式识别方法,是Vapnik等提出的基于结构风险最小化原则(structural risk minimization pnneiple,SRM)的统计学理论,用于分类与回归问题,并给出了核心概念VC(vapnik-chervonenkis dimension),指出了最小化期望风险必须同时最小化经验风险和VC维数。SRM使VC(vapnik cherovnenkis)维数的上限最小化,这使得SVM方法比基于经验风险最小化的人工神经网方法具有更好的泛化能力。

直接使用SVM方法进行人脸识别有两个困难:

(1) 训练SVM需要求解二次规划问题,计算复杂度高,内存需求量巨大。

(2) 训练非人脸需要大量样本,使得分类器的计算量过高。

J. C Plait提出了称为顺序最小最优化(sequential minimal optimization,SMO)的支持向量机训练方法,将一个大型的求解二次规划问题变成一系列最小可能的二次规划问题,这样就避免对大型的二次规划问题直接求解,有效地解决了第一个问题。Ostma等人首先将SVM方法用于人脸检测问题,在训练中使用了大量人脸样本,采用“自举”的方法收集“非人脸”样本,并使用逼近优化的方法减少支持向量的数量,在一定程度上解决了第二个问题。梁路宏等采用模板匹配与SVM方法相结合的人脸检测算法,在模板匹配限定的子空间内采用“自举”的方法收集“非人脸”样本,训练SVM,降低了训练的难度和最终得到的支持向量的规模,使得检测速度比单纯的SVM检测器提高了20倍,得到了与CMU的神经网络方法可比较的结果。

6.2.4 人脸识别技术概述

人脸识别(face recognisation)技术的核心内容是从已知人脸来确定未知人脸的归属问题。在数据库里预先存在若干已知人脸图像或相关特征的前提下,将待识别图像的特征和库里图像的特征进行匹配,使得待识别人脸与库中已存在的某个人脸对应起来确定待测身份。

由于自动人脸识别技术在安全控制和人机交互等领域内有广泛的应用,因此人脸识别成为近三十年来模式识别和图像理解中最热门的研究主题之一。人脸识别算法主要是基于几何特征或模板的匹配。基于几何特征匹配的人脸识别方法利用面部特征点的大小、位置、距离、角度、形状等参数作为特征进行人脸识别。模板匹配法利用相关匹配比较待识别图像和标准模板,从而识别人脸。

经过多年的研究,逐渐形成了人脸识别的几个主流研究方向:基于主元分析的特征脸法(eigen face)、基于 Fisher 线性判断分析的 Fisher 脸法、弹性图匹配法(elastic graph matching)以及局部特征分析法(local feature analysis,LFA)。特征脸技术作为一种比较成功的人脸识别技术,掀起了人脸识别研究的第二次高潮。Fisher 脸方法在特征脸方法的基础上,引入类内共性和类间差异的分类信息,使得投影子空间用于分类问题。弹性图匹配法是一种解决脸相多姿态变化问题的基于局部信息的人脸识别方法。局部特征分析方法考虑了面部局部特征的信息以及它们之间的拓扑关系。

除此之外,从分类器的角度出发,许多研究采用了基于神经网络的方法、基于支持向量机的方法、基于多分类器的方法等。另外隐马尔可夫模型也被用于人脸识别。除了上述对于静态图像人脸识别的研究,三维图像识别研究和三维动态脸相识别也是人脸识别研究的重要组成部分。

6.2.5 基于特征的识别算法

1. 基于面部几何特征的方法

几何特征方法采用的特征包括人脸五官如眼睛、鼻子和嘴巴等局部形状特征,脸型特征以及五官在脸上分布的几何结构特征。提取特征时往往要用到脸部结构的一些先验知识,早期采用从侧面轮廓线提取特征的方法比较多,以后的研究工作较多地采用人脸的正面特征,因为证件照大多数是正面照,而且正面的特征信息比侧面的更丰富,特征的抗干扰能力也比较强。几何特征法用几何特征矢量表示人脸,用模式识别中层次聚类的思想设计分类器达到识别的目的。几何特征矢量是以人脸器官的形状和几何关系为基础的特征矢量,分量通常包括人脸指定两点间的欧氏距离、曲率和角度等参数。选取的几何特征矢量需要有一定的独特性,能够反映不同人脸之间的差别,同时又具有一定的弹性,以消除时间跨度、光照等的影响。

Kanade 设计的系统提取了 16 个正面人脸几何特征,使用的数据库中每个测试者有一个训练图像和一个测试图像,共 20 个测试者,达到了 75%的识别率。Brunelli 和 Poggio 用改进的积分投影法提取出用欧氏距离表征的 35 维人脸特征矢量。积分投影方式产生出的波峰波谷分别对应于不同的人脸器官关键点,根据人脸结构的先验知识,可以得到人脸器官之间的几何位置关系。该识别器在一个有 47 个被试者的数据库上测试,获得了大约 90%

的识别率。Goldstein 等人使用了包含 34 个既有前视图又有侧视图特征的集合。这些特征包括头发长度、头发纹理、鼻子长度、嘴巴宽度和下巴轮廓特征。该模型也取得了良好的识别率。基于几何特征的方法比较符合人类的认知机理，易于理解，且图片对光照的变化不敏感，但是从图像中抽取稳定的特征比较困难，特别是特征受到遮挡时。强烈的表情变化和姿势变化对算法的影响也比较大。此外，一般的几何特征只描述了部件的基本形状与结构关系，忽略了局部细微特征，会造成部分信息丢失。

2. 基于小波包分解的局部特征的识别方法

小波变换是国际上公认的最新频率分析工具，由于其“自适应性”和“数学显微镜性”而成为许多学科共同关注的焦点，在信号处理中起着至关重要的作用。小波变换采用以高斯函数的二阶导数作为小波基来进行拐点提取，然后以该方法为基础，进行不同图像之间拐点序列的匹配；最后再利用提取的拐点来对图像进行分段和段与段对应处理。由于使用离散小波变换来分解图像的参数特征，特征提取要用到自适应算法。而特征匹配则选择动态规划方法。

算法中采用 Haar 小波作为基函数。由于在实验中所采用的面部特征区域比较小（21×21），因此二级的分解就能够获得丰富的细节。利用二级分解得到的各个系数矩阵，可以重建出 16 个图像 $I_0 I_1 \cdots I_{15}$。这些图像的能量可以通过公式进行计算。

$$A_k = \iint I_k^2(x,y)\mathrm{d}x\mathrm{d}y$$

其中 $k=0,1,\cdots,15$。向量 $A=(A_0,A_1,\cdots,A_{15})$作为所提取的面部局部区域的特征，反映了图像 I 在不同尺度的小波频段上的能量尺度。

3. 基于弹性图匹配的人脸识别算法

弹性匹配（elastic matching）属于模板匹配，是动态匹配的一种。通常的模板匹配方法用网格作为模板，首先抽取图像模板，然后将两幅图的模板进行比较。弹性匹配方法也用网格作为模板，将图像间的比较转化为网格间的比较。对于网格上的每一点抽取一定的特征信息，如灰度值、梯度值、傅里叶变换系数值、小波变换系数值等，形成一组特征矢量，并用这些特征矢量来代表图像特征进行匹配。根据要求不同，可以抽取不同的特征，利用弹性匹配法进行特征比较和识别。目前国内外提出了多种基于不同特征的弹性匹配人脸识别方法，其中比较著名且效果较好的是 Lades 等人提出的一种基于动态链接结构（dynamic link architecture，DLA）的弹性图匹配方法。在这个方法中，他们将人脸表示成图，图中的节点是一些基准点（如眼睛、鼻尖等），图中的边为这些基准点之间的连接。每个节点包含 40 个 Gabor 小波系数，包括相位和幅度，用来表征人脸图像。

基于 DLA 的弹性图匹配方法利用 Gabor 小波变换来描述局部特征点的信息，能较好地反映人脸图像局部区域的多尺度信息，同时保留图像像素的空间相似性，因此利用 Gabor 小波滤波器抽取的特征能克服光照、尺度、角度等全局干扰对识别效果的影响，同时由于采用了可变形的人脸模板，这种方法能克服表情的细微变化对识别效果的干扰。另外，当向图像库中加入新的图像时，无须改变已有的特征库，这样操作起来比较方便。然而，弹性匹配法计算量非常巨大，需要消耗大量的时间。其次没有考虑到人脸各个不同位置在识别时所起的不同作用，所有节点都相同处理，影响了识别效果。此外由于受网格密度的影响，该方法只能对表情细微变化保持一定的不变性。

6.2.6 基于子空间分析识别方法

由于人脸图像的维数通常都是很高的，而实际上人脸图像在这样的高维空间中分布很不紧凑，不利于分类，并且在计算上的复杂度也非常大，因此人们往往将人脸图像投影到低维的子空间进行判别。子空间方法的基本出发点是根据一定的优化目标来寻找线性或非线性的空间变换，把原始信号数据压缩到一个低维的子空间中，使数据在该子空间的分布更加紧凑，为数据的描述提供了更好的手段，其计算复杂度也大为降低。如果把人脸图像看成整个空间的话，那么每一个人脸样本都是分布在这个空间中的一个点。可以考虑经过一个映射，将这些高维人脸空间中的点投影到另一个低维空间中。由于子空间的获取需要求助于一个投影，所以基于子空间的方法也被称为基于投影的方法(projection based methods)。

1. 基于主成分分析的方法

主成分分析(principal component analysis，PCA)方法将人脸图像看成是随机向量，通过图像向量化进行 K-L 变换获得其正交基，再通过保留主成分，得到低维的人脸向量空间。每个待识别图像和训练样本均可以用该向量空间中的一点表示，通过计算它们之间的欧氏距离即可进行识别。

完整的人脸识别算法步骤如下：

(1) 读入人脸库。

(2) 计算 K-L 变换的生成矩阵。

(3) 利用 SVD(奇异值分解)定理计算图像的特征值和特征向量。

(4) 把训练图像和测试图像投影到特征空间。

(5) 比较测试图像和训练图像，确定待识别样本类别。

归一化人脸库后，将库中的每人选择一定数量的图像构成训练集，其余构成测试集。设归一化后的图像是 $n\times n$，按列相连就构成 n^2 维矢量，一幅图像可视为 n^2 维空间中的一个点 x_1，可以通过 K-L 变换用一个低维子空间描述这个图像。

以训练样本集的总体散布矩阵为产生矩阵，即

$$\sum = E\{(x-\mu)(x-\mu)^{\mathrm{T}}\}$$

也就是

$$\sum = \frac{1}{M}\sum_{i=0}^{M-1}(x-\mu)(x-\mu)^{\mathrm{T}} = \frac{1}{M}R$$

式中 x_i 为第 i 个训练样本的图像向量；μ 为训练样本均值向量；M 为训练样的总数。为了求矩阵的特征值和正交归一的特征向量，直接计算的话计算量太大，可引入奇异值分解定理来解决维数过高的问题。

先计算出各个特征值 α，把特征值按大小排序。计算出前 M 个特征值对应原始数据正交的特征向量 W。将原始数据在特征向量 W 上进行投影，即可获得原始图像的特征数据。

2. 基于核主成分分析与核线性判别分析的方法

基于核(kernel-based)的方法是当前模式识别领域中一个发展迅猛的新方向。它的主要思想最初由 Vapnik 提出并应用于支持向量机的方法中。Scholkopf 等人将核方法应用于特征提取中，提出了核主元分析方法(kernel PCA，KPCA)。与 PCA 相比，KPCA 能够提取非线性特征，并且能达到更高的识别率。Mika 等人利用核方法将 LDA 进一步推广，提出

了核线性鉴别分析法(KLDA)。实验结果表明 KLDA 抽取的是高维特征空间中最具分类能力的特征,本质上是输入空间中非线性的最具分类能力的特征。

对于给定非线性映射 Φ,将输入空间 R^N 映射成特征空间 F:

$$\Phi: R^N \rightarrow F$$

一个在 R^N 的模式映射特征空间中具有更高维数的模式,即在 R^N 空间中线性不可分的模式,在映射后的特征空间中将可能变得线性可分,或者是比在 R^N 空间中更容易分类。KPCA 就是一种在特征空间中进行 PCA 的方法。

许多实际问题往往是线性不可分的。非线性方法往往可以有效地进行特征表示和分类。核主成分分析和核线性判别分析通过非线性映射把低维空间的样本非线性映射到高维空间,从而得到更好的表示,然后在高维空间中用线性方法提取特征。因此,可以认为它们在样本空间中实现了非线性的特征提取。在实际计算中,引入核函数使得计算简便有效。

6.3 虹膜识别技术

6.3.1 概述

虹膜作为重要的身份鉴别特征,具有唯一性、稳定性、可采集性、非侵犯性等优点。而虹膜识别技术作为近年来新兴起的一种生物识别技术被越来越广泛地应用于金融业、保险业、电子商务等需要进行身份认证的行业。据统计,到目前为止,虹膜识别的错误率是各种生物识别中最低的,其重要价值逐渐凸显。

眼睛由巩膜、虹膜、瞳孔三部分组成,如图 6-14 所示。其中,巩膜即眼球外围的白色部分;眼睛中心为瞳孔部分;虹膜位于巩膜和瞳孔之间,包含了丰富的纹理信息。虹膜包含

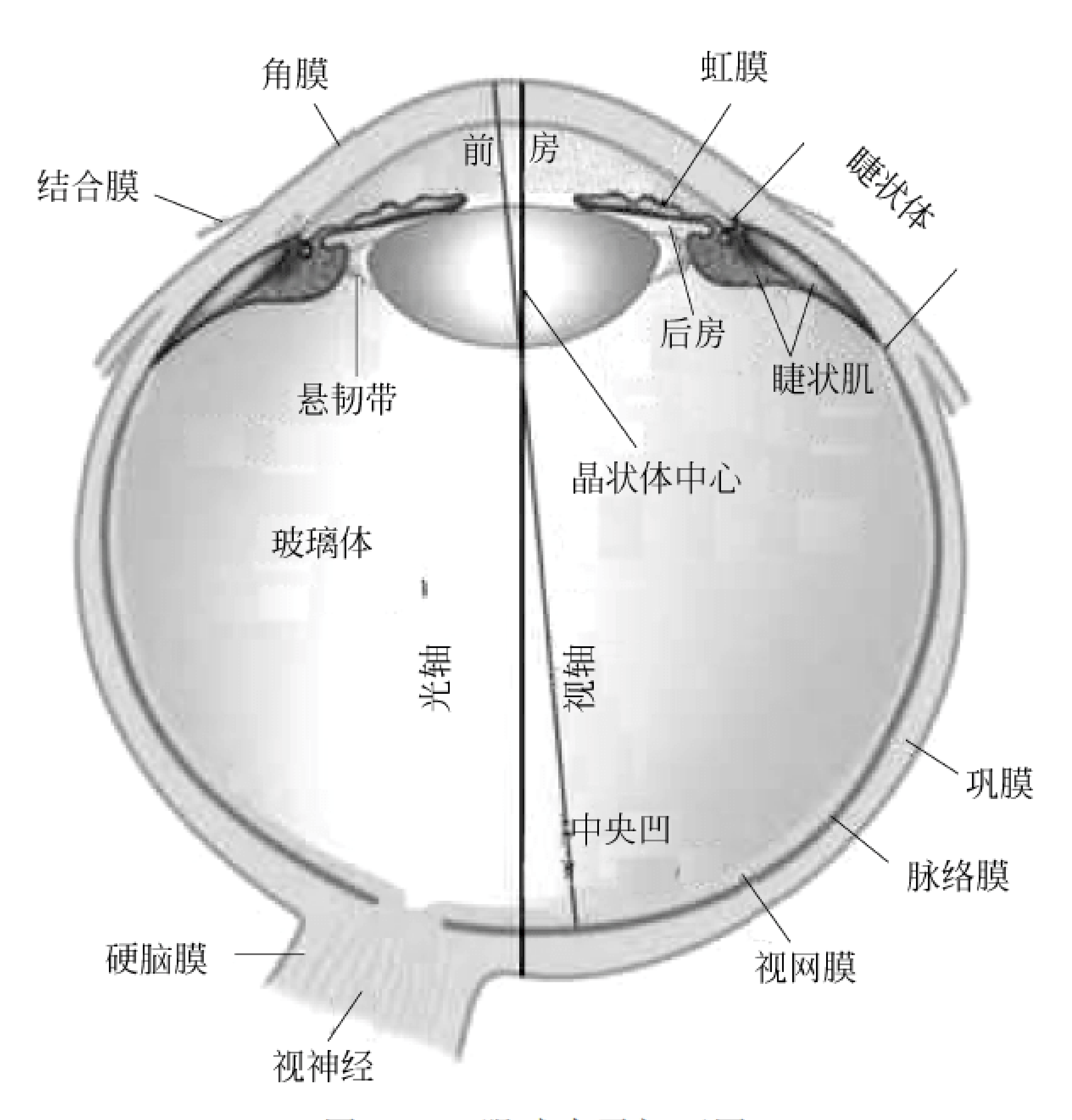


图 6-14 眼球水平切面图

有丰富的色素细胞，当外部光线照射到眼睛上时，由于不同人的色素细胞对光有不同的吸收率，使得虹膜呈现不同的颜色。从识别的角度来说，这些相互交错的类似于斑点、细丝、冠状、条纹、隐窝等形状的细微特征是虹膜唯一性的体现。虹膜的高度独特性、稳定性及不可更改的特点，是虹膜可用作身份鉴别的物质基础。

一个自动虹膜识别系统一般包含硬件和软件两大模块：虹膜图像获取装置和虹膜识别算法，分别对应于图像获取和模式匹配这两个基本问题。自动虹膜识别系统的工作流程如图 6-15 所示。

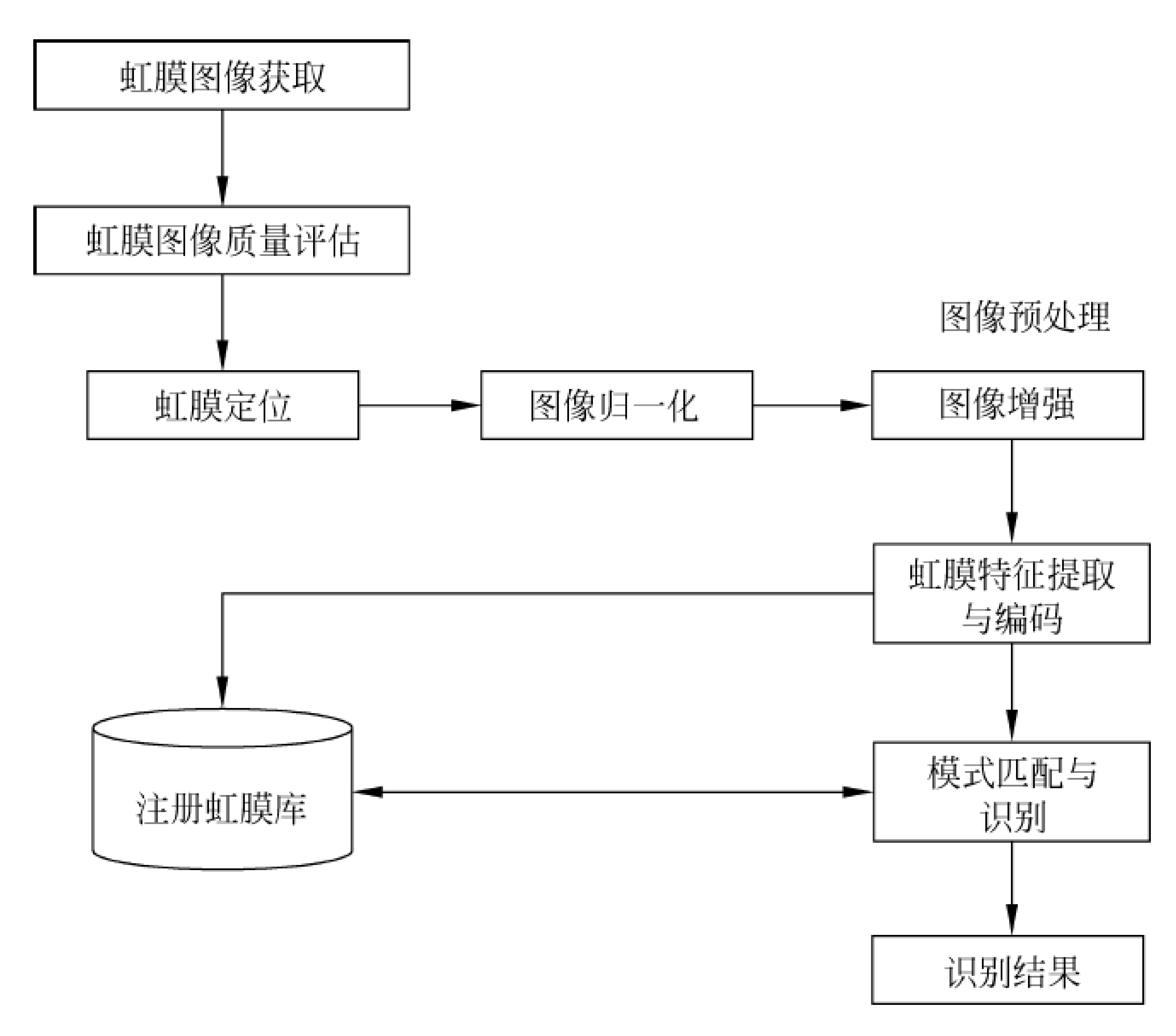


图 6-15 虹膜识别系统的工作流程图

6.3.2 虹膜图像获取

虹膜识别的关键一步就是获取高质量的虹膜图像。由于虹膜的区分主要在于纹理细节的不同，而对于东亚人种来说，虹膜的纹理不是很清晰。用普通的 CCD 摄像头和在正常的光照条件下很难获得清晰的虹膜图像。因此，需要设计带有红外光源的虹膜图像采集装置。当前的虹膜采集设备的制作原理主要有两种：美国 Wildes 博士提出的光线散射式(原理如图 6-16 所示)和英国剑桥大学的 Daugman 博士提出的光线折射式(原理如图 6-17 所示)。

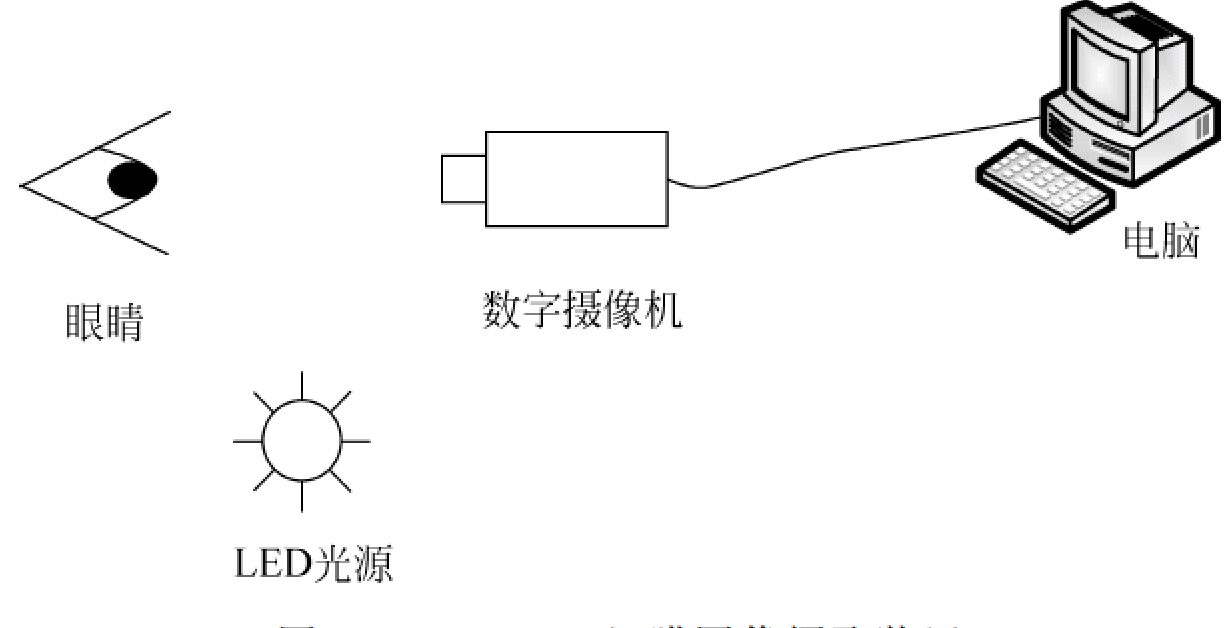


图 6-16 Wildes 虹膜图像摄取装置

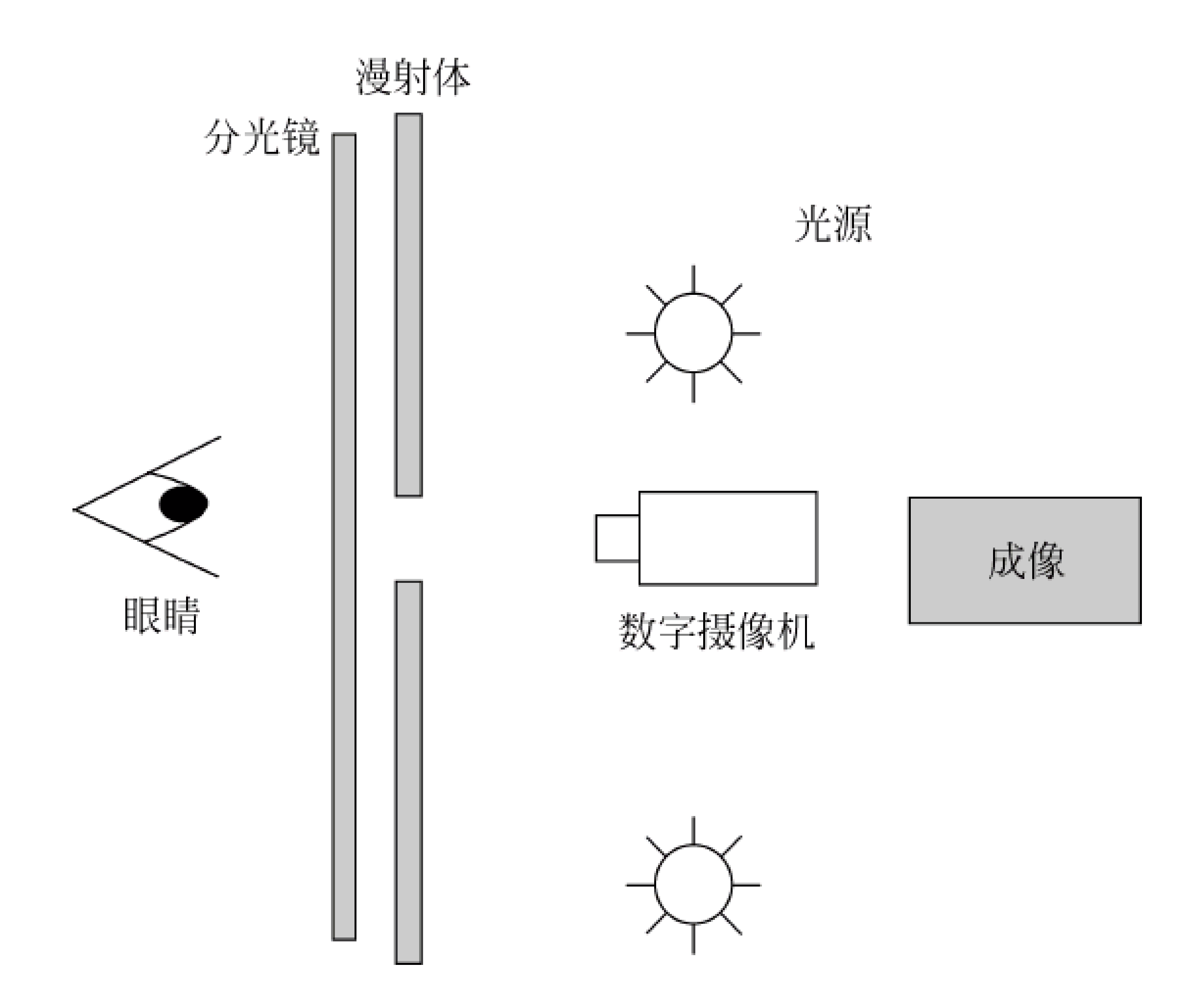


图 6-17 Daugman 虹膜图像摄取装置

散射式虹膜图像摄取装置难以获得纹理不清晰的虹膜图像，因此已逐渐被淘汰，其原理不再详细描述。

Daugman 博士提出的光线折射式虹膜摄像装置是将该装置外壳的上端扣在被测者的眼睛上，在红外发射管及发光二极管的照射下，虹膜图像被 CCD 摄像头采集并输入到计算机中。光线折射式虹膜摄像装置原理：数字摄像机从 30～50cm 的距离摄得分辨率为 256×256 像素的眼部图像，为避免外界杂光的干扰，采集环境设计成封闭型，同时考虑到人眼的舒适度要求，采用红外 LED 按某种方式布置在圆周上，利用透射光对虹膜进行照明，并用发光二极管作为采集时的人眼定位指示标志，以最大限度减弱留在虹膜上的光斑，获得尽可能多的虹膜细节。

6.3.3 虹膜定位

常用的定位方法主要有阈值法、Hough 变换、积分微分算子、基于几何特征的方法、主动轮廓线法等。若从虹膜的结构角度分，又可分为瞳孔定位和外圆定位两个步骤。

1. 瞳孔定位

首先介绍一下瞳孔定位。以红外光源拍摄的虹膜图像瞳孔的边缘会更明显。因此通常的虹膜定位算法为了提高效率会先定位瞳孔，也就是虹膜的内边缘，然后以瞳孔的圆心为基准再定位外边缘。瞳孔的准确定位对整个虹膜的定位来说至关重要。本节对瞳孔的定位分为粗定位、修正以及拟合 3 个步骤来完成。

(1) 粗定位

粗定位的目的是先从整幅图像中定位出大致的瞳孔区域，主要是根据瞳孔区域的灰度分布特征，利用数学形态学的方法来进行的，大概步骤如下：

第一步：首先根据瞳孔的灰度信息进行二值化处理。阈值的选取根据图像的直方图信息，第一个跳变较大的谷值对应大致的瞳孔区域。为了避免灰度的突变引起的阈值搜索错误，首先将直方图进行平滑处理，然后找出第一个明显的谷值，作为阈值将原图像二值化。应该注意的是，为了避免眼窝、眼睑、眉毛等容易在图像边缘位置出现的大块阴影，应该限制

在大致图像中间的位置搜索，如图 6-18(b)所示。

第二步：在二值化后的图像中，找出最大的连通区域，即为瞳孔所在的区域，结果如图 6-18(c)所示。但其中包括部分粘连的颜色较深的睫毛及眼睑。

第三步：对该区域进行开运算后分离出其中最大的连通区域，可将部分连接的睫毛、眼睑分离出去，结果如图 6-18(d)所示。

第四步：搜索该区域中是否存在光斑。若存在则将其填满，结果如图 6-18(e)所示。

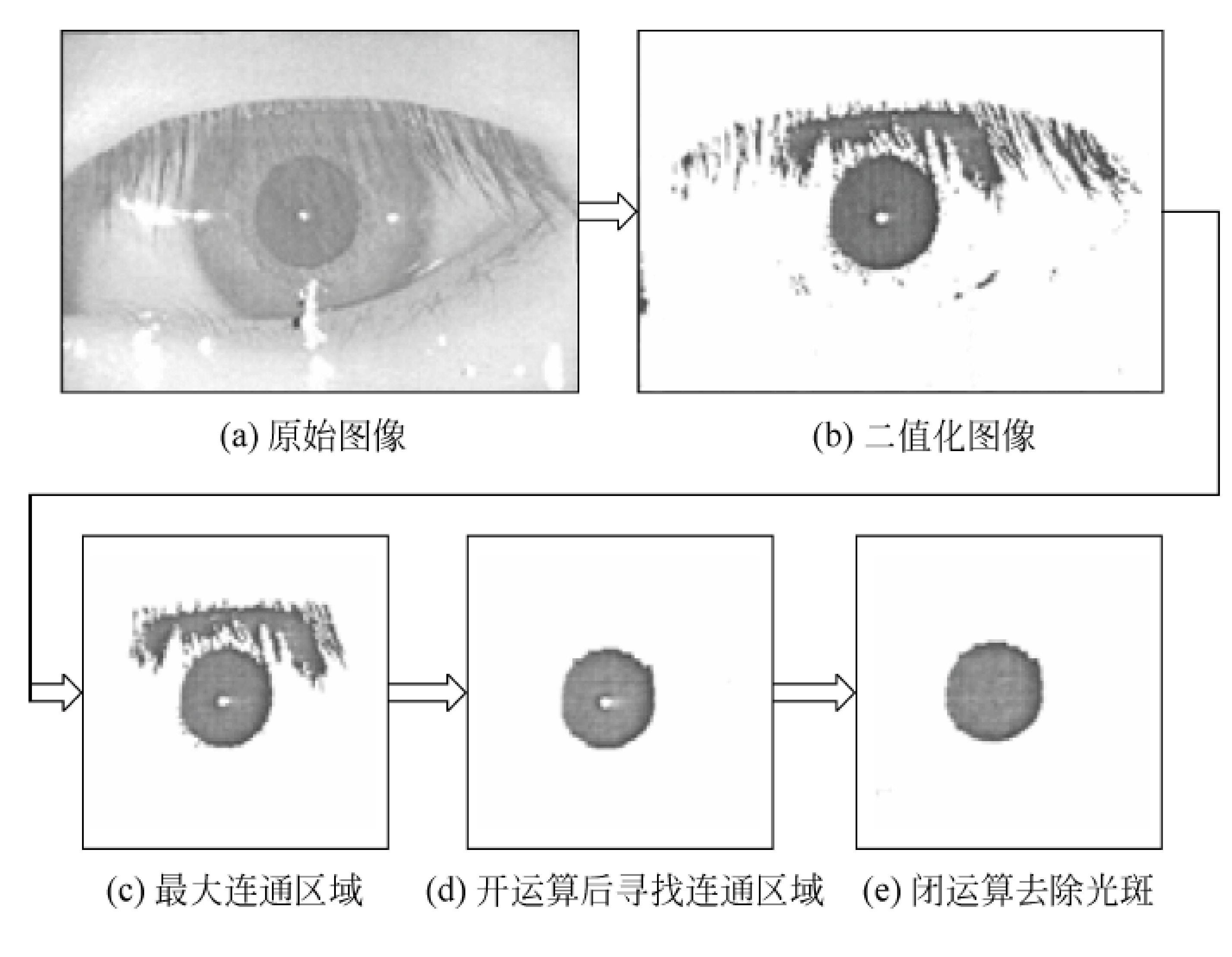


图 6-18 **瞳孔定位**

可以看出，进行完此操作，通常可得到较准确的瞳孔区域。但在很多情况下，进行完以上步骤后，仍不能得到较好的结果，所以需要进行下一步的修正处理。

(2) 修正

在部分图像质量较差的情况下(主要是遮挡严重)，进行完上述操作后，可能仍存在大量的睫毛、眼睑与瞳孔部分相粘连，如果直接进行外圆的拟合，势必会造成参数的较大偏差。另外粗定位中如果阈值的选取并不是非常恰当，也会出现结果偏差较大，所以需要进行修正。

为了使计算机能够识别出大概的瞳孔区域，需要进行如下的修正步骤。

第一步：首先确定图像中阴影部分的最高位置和最低位置，这样就可以得到中间线位置。

第二步：确定图像中阴影部分的最左点和最右点，然后得到两线之间的中间线位置。第一、二两步所得两线交点即为瞳孔中心点。

第三步：去除上下边界和左右边界之外的区域即得到了瞳孔的有效区域。

由于采用四线来确定瞳孔边界，所以修正后的瞳孔区域可能被确定在一个矩形区域内，而不是我们通常认为的圆形区域。因此仍需要进一步的拟合操作才能得到一个确切的圆形瞳孔边界。

(3) 拟合

根据瞳孔的生理形状特征，可以用圆形来拟合瞳孔。根据以上的方法将大致的瞳孔区

域分割出来后，需要进一步拟合操作才能得到更精准的圆形瞳孔区域。

本书采用 Chauduri-Kundu 提出的基于最小均方误差的参数确定方法。均方误差 J 为

$$J = \sum \omega_i [(x_i - x_{\text{pupil}})^2 + (y_i - y_{\text{pupil}})^2 - R_{\text{pupil}}^2]^2$$

其中(x_i, y_i)是利用 Sobel 算子得到的次区域的边缘点集，$(x_{\text{pupil}}, y_{\text{pupil}})$为虹膜外圆的中心位置，$R_{\text{pupil}}$为虹膜外圆的半径。

根据最小均方差原则，x_{pupil}，y_{pupil}和 R_{pupil}的值可以由下列公式迭代计算出。

$$x_{\text{pupil}} = \frac{B_y C_x - B_x C_y}{A_x B_y - A_y B_x}$$

$$y_{\text{pupil}} = \frac{A_y C_x - A_x C_y}{A_y B_x - A_x B_y}$$

$$R_{\text{pupil}}^2 = \frac{1}{W} \sum \omega_i [(x_i - x_0)^2 + (y_i - y_0)^2]$$

其中：

$$W = \sum w_i$$

$$\bar{x} = \frac{1}{W} \sum \omega_i x_i$$

$$\bar{y} = \frac{1}{W} \sum \omega_i y_i$$

$$A_x = \sum \omega_i (x_i - \bar{x}) x_i$$

$$B_x = \sum \omega_i (x_i - \bar{x}) y_i$$

$$C_x = \frac{1}{2} \sum \omega_i (x_i - \bar{x})(x_i^2 - y_i^2)$$

$$A_y = \sum \omega_i (y_i - \bar{y}) x_i$$

$$B_y = \sum \omega_i (y_i - \bar{y}) y_i$$

$$C_y = \frac{1}{2} \sum \omega_i (y_i - \bar{y})(x_i^2 - y_i^2)$$

即使是经过修正的瞳孔区域，仍然会受到其他因素的影响。但是可以通过在拟合中调节参数 ω_i 的值来控制，也就是将干扰的边缘值赋予较低的 ω_i 值。但是在初始状态并不知道哪些边缘点是真正的瞳孔边缘点，哪些点是干扰点，本书将初始的 ω_i 设为相同的值，即 $\omega_i=1$。然后在迭代拟合的过程中，通过各点与此步所拟合的圆的偏差来调整 ω_i 值，进而调整 x_{pupil}、y_{pupil}的值。实验证明，迭代 10 次，就能很好地收敛到瞳孔的位置。

2. 外圆定位

在进行瞳孔的定位后，就要进行外圆的定位。虹膜外边缘由于受到眼睑、睫毛等噪声的干扰，且虹膜与巩膜分界不明显，定位难度比瞳孔定位难度要大，所以在边缘抽取时不能用简单的算子进行操作。另外，由于虹膜外边缘受噪声干扰大，边缘抽取后也不能简单地用几何的方法定位虹膜外圆。常用的方法有 Hough 变化法和微分积分算子。考虑到使用 Hough 变化对边缘检测的结果较敏感，并且计算量较大，本书采用微分积分算子来确定外圆的参数。

首先使用高斯低通滤波器 $G(i,j)$ 对图像进行平滑处理。

$$G(i,j)=\frac{1}{\sum_{i}\sum_{j}G_f}G_f(i,j)$$

$$G_f=\mathrm{e}^{\frac{-(i^2+j^2)}{2\sigma^2}}$$

这样使虹膜图像的外边界上尽可能多的像素点的灰度值趋于一致，同时又不至于消除内外的灰度差异。接下来使用微分积分算子的离散形式来进行圆的检测，可由以下公式表示

$$\max_{(n\Delta rx,y)}=\left|\frac{1}{\Delta r}\sum_{k}[G_\sigma(r)\sum_{m}I(x,y)]\right|$$

其中：$G_\sigma(r)=G_\sigma((n-k)\Delta r)-G_\sigma((n-k-1)\Delta r)$；$\Delta r$ 表示半径搜索的步长。

另外，考虑到一般情况下，虽然瞳孔与虹膜外圆并不同心，但其偏差并不大。所以根据瞳孔定位的结果，虹膜外圆的中心 $(x_{\text{iris}},y_{\text{iris}})$ 在 $(x_{\text{pupil}}\pm5,y_{\text{pupil}}\pm5)$ 范围内搜索，避免了直接对原始图像进行操作，降低算法的复杂度（但同时也限制了瞳孔的定位精度，在一定程度上决定了虹膜外圆的定位效果，决定了虹膜定位的成败）。为了进一步降低算法的复杂度，也可以将虹膜图像按一定比率缩小后（例如缩小为原始图像的 50%）再进行外圆的搜索。

6.3.4 图像预处理

由于设备的原因，虹膜图像上的光照不能完全均匀分布，这样将会影响纹理分析的效果，为了更好地提高识别效果，一般需要对展开的虹膜图像进行局部的直方图均衡化，从而实现图像增强，减少非均匀光照的影响。

该方法所使用的变换函数为灰度级累积分布函数 $T(r)$，设原始图像的灰度级为 r_k，$S(r)$ 为变换后的图像的灰度分布函数，N 为图像中的像素总数，$N(r_i)$ 为图像中灰度级为 r_i 的像素的总数，则

$$S(r)=T(r_k)=\frac{1}{N}\sum_{i=0}^{k}N(r_i)$$

这样可得到一个近似均匀分布的直方图分布函数 $S(r_k)$，与原始图像相比，它的灰度分布值得到了扩展。

除了用局部的直方图均衡化方法来实现图像增强外，基于重建的超分辨率方法也成为了虹膜图像增强领域的另一个重要分支。该方法是利用统一景物重复拍照所获取的多帧图像的冗余信息，消除和降低混叠效应，重构出超分辨率图像。通常情况下，基于重建的超分辨率方法在进行多帧图像融合时，会加入一些限制条件，使结果图像要满足低分辨率图像的局部基本特性。下面简单介绍一些基于重建的虹膜图像增强方法。

1. 频域方法

这种方法由 Huang 和 Tsai 于 1984 年首次提出，该方法假定帧间运动仅为平移运动，且运动关系已知，即已获得亚像素级的物体运动信息，同时假设图像在频域中的带宽是有限的。设连续图像为 $f(x,y)$，考虑 K 幅平移图像

$$f_r=f(x+\Delta x_r,y+\Delta y_r)$$

其中，$r=1, 2, \cdots, K$。

对上述函数进行重新采样，得到

$$y_r(m,n) = f(mT_x + \Delta x_r, nT_y + \Delta y_r)$$

其中，$m=1, 2, \cdots, M-1$；$n=1, 2, \cdots, N-1$；T_x 和 T_y 分别为 X 轴和 Y 轴的采样周期。第 K 幅图像的离散傅里叶变换为$Y_r(k,l)$，连续与离散傅里叶变换关系是

$$Y_r(u,v) = \alpha \sum \sum F_r\left(\frac{u}{MT_x} + pf_x, \frac{v}{NT_y} + qf_y\right)$$

其中 $f_x=\frac{1}{T_x}$，$f_y=\frac{1}{T_y}$分别为 X 轴和 Y 轴上的采样频率；$\alpha=\frac{1}{T_xT_y}$。由傅里叶变换的平移性可得到

$$Y = \phi F$$

这样，若已得到 N 帧图像的傅里叶变换，且已知这 N 帧图像的平移变化，同时由这 N 帧图像得到的矩阵 ϕ 为满秩的，则可得到所需的高分辨率图像。

2. 插值方法

插值方法就是对图像采样点进行扩充。按照不同的思路，插值方法又分为两类。

一类是多帧图像对应值填充法。因为各帧图像中的物体总会发生扭曲，所以这种方法不仅需要进行运动估计，而且需要考虑不均匀采样，通过循环的方法利用各帧图像信息来达到融合的主要目标。

另一类方法称为细分法，这种方法通过对图像粗格点进行充分的迭代采样，利用数学工具考察各种迭代方法的收敛性和收敛速度，最重要的是它拥有能够得到更低阶连续图像的能力，这样就可避免样条插值那样引起边缘被平滑。

有人提出了根据图像局部特征确定插值策略的方法，如根据局部协方差、边缘方向等。插值方法是处理平滑区域最直接最有效的方法，如果能有效地把运动估计和平滑区域划分与插值方法结合，将会有较好的效果。

3. 水平线方法

很早就有人提出利用水平线的思想，即对图像中连通的灰度区域进行统一编码。虽然由于图像中噪声、复杂纹理等现象的存在，使得研究者很难得到有效的算法，但水平线方法不失为一种对图像有效分割的好思路。Froment 和 Whitaker 等人经过多年的努力，已经解决了水平线方法应用中的一部分难题，如：

- 提出多层次的灰度阈值选取思想。
- 指出在图像中主要水平线的决定因素。
- 提出不同水平线间区域的填充算法。
- 提出对于纹理区域的处理方法。
- 提出相邻图像帧间水平线融合方法。
- 提出利用水平线方法进行图像压缩的框架。

超分辨率问题的主要困难在于不同性质区域的处理和信息的多层次性，水平线方法可能会成为部分解决这些问题的思路，但总体而言，这些解决方法还处于理论研究阶段。

6.3.5 虹膜识别算法

虹膜识别算法大致可分为 3 类。下面对这 3 类算法进行简单介绍。

1. 基于特征点的方法

这是一种基本的匹配方法，通常包括三个步骤：首先提取图像的特征点，然后将两幅图像中的特征点对应起来，最后根据对应的特征点确定空间变换，通常是两个二维多项式。在这种方法中控制点数量、位置的选择及特征点匹配的精确度起着重要的作用，它们直接影响着匹配的精确性，这是因为在完成特征点的配准之后，剩余的工作就仅仅是插值或逼近了。直接基于特征点的方法发展历史较长，比较灵活，运算量也相对较小，但从整体上来说精度不高。

2. 基于分割的方法

这类方法包括两个步骤，首先提取出两幅图像中对应的曲线或曲面，然后再根据这些对应的曲线或者曲面决定几何变换。变换的形式既可以是刚体变换也可以是形变变换。以形变变换为例，目前基于形变的模型通常是采用弹性形变的方式，作用于分割后的曲线或者曲面上，并采用迭代的方式逐渐完成。形变的曲线一般称之为 snakes 或 active cotours，在三维中称为 nets。采用这种方法首先要在一幅图中提取出一个模板模型，模板不停形变直到和另一幅图像中分割提取出的几何结构相匹配，而另一幅图像不进行分割，这时模板曲线变换到另一图像中某区域的边缘位置。这种方法在初始曲线和目标曲线差别较大时效果不好，这时可采用刚体变化的方法进行预匹配，然后再进行形变匹配。

3. 基于纹理分析的方法

基于纹理分析的方法和上述两种不同，它直接作用于图像中的像素而不需要进行数据缩减或分割。它利用图像统计信息作为配准依据，也被称为矩和主轴法。它借用经典力学中物体质量分布的概念，计算两幅图像像素的质心和主轴，再通过平移和旋转使得两幅图像的质心和主轴对齐，从而达到匹配的目的。矩和主轴对数据的缺失较敏感，要求整个物体必须完整地出现在两幅图像中。

6.4 掌形识别技术

6.4.1 概述

随着科学技术的进步，各种生物特征识别技术都得到了充分的发展。其中掌形识别与指纹识别、人脸识别以及虹膜识别一样作为一种新型的身份认证方法被越来越多的应用。掌形识别技术包括掌纹识别技术和手形识别技术，掌纹是指手掌上的纹理特性，而手形指的是手的几何特征，例如手掌的宽度、厚度，手指的长度、宽度等。掌形与其他生物特征一样，也具有稳定性、唯一性和普遍性，因此也可以作为身份确认的识别特征。近年来掌形识别技术的应用和研究已经取得了很大的进展，逐渐被人们接受和认可。

目前已有的掌形识别商业化产品主要是采用手形识别技术，比如美国英格索公共所的 Handkey 掌形仪，如图 6-19 所示。香港理工大学生物特征研究中心在掌纹识别技术研究方面也做出了突出的贡献。本节主要介绍手形识别技术。



图 6-19 Handkey 掌形仪

6.4.2 特征采集及预处理

手形特征是指手的几何结构,这一结构包括不同位置的手指宽度、手掌的宽度、手掌的厚度以及手指的长度等。尽管这些特征在不同个体间有很多相似之处,但还是能使用它们来区分个体。一般的手形图像采集系统包括一个光源、一个相机、一个单面镜和一个平板。采集图像时用户将手掌朝下放在平板上,有 5 个小柱子用于控制用户放置手的合适姿势,这个装置还能改变光源的强度以及相机的焦距。单面镜的作用是将手掌的一个侧面投影到相机上,这样可以采集到诸如手掌、手指的厚度等几何信息。目前的掌形识别系统还没有利用到手的非几何特征,如皮肤的颜色等。

采集到手形图像后的一个重要的步骤就是手形图像的预处理,其目的在于将手的形状从采集图像中完整地分离出来。与指纹图像相比较,手形图像的预处理要简单一些。主要步骤如下。

(1) 图像平滑和灰度调整。利用滤波器对原始图像进行平滑滤波处理,再将图像归一化,使得图像的灰度值具有预先设定的均值和方差。

(2) 图像二值化。根据预先指定的阈值将图像进行二值化处理。

(3) 图像边缘检测和轮廓跟踪。在二值化图像上利用边缘提取算法提取出人手边缘,再利用轮廓跟踪算法提取完整的人手轮廓。

6.4.3 掌形识别算法

手形特征提取快速准确,而且手形识别算法的特征模板小,匹配速度快。匹配算法也多

种多样，比如基于左右半轴宽度匹配识别算法、基于单手指匹配识别算法、基于特征向量识别算法、基于点模式识别算法等。下面简单介绍两种较为常用的手形识别方法。

1. 基于手形尺寸测量值的识别算法

首先利用边缘拟合方法确定每个手指较精确的起始点和终止点，然后根据每个手指的宽度线方向，对各个手指分别做宽度线。若将宽度线从指尖滑向指根，就得到了手指的一系列边缘点和宽度值，从而形成手指的宽度变化曲线。根据宽度变化曲线可以确定指尖的范围。确定指尖的目的是为了排除手指甲的影响。确定每个手指的形心，再以形心为原点对每个分离的手指坐标变换，得到每个手指的相对坐标形式。

比对时，采取每个手指分别匹配的方法。以一个手指为例：选择两指前点坐标和宽度最为详尽的点分别为用户和模板的手指起始端点。然后对用户手指和模板手指从起始端到终止端平均选取 100 个宽度值，同时对应了 200 个手指边缘特征点。首先计算用户某一部位的宽度与模板相应部位宽度的差，然后设定阈值，当宽度差小于该阈值时，认为用户和模板在该部位的宽度相匹配，否则认为不匹配。这样，就求得用户与模板相匹配的宽度数及百分比率。当有超过 80％的宽度相匹配时，则认为用户手指和模板手指相符的。当且超过 3 个手指都相符时，系统认为该用户与模板属于同一个人。

2. 基于高斯混合模型的识别方法

这种方法的核心思想是将手形图像表示为一个由轮廓点构成的可观测序列，在这种情况下，如果能够建立一个模型来描述这个序列的特征，那么就可以利用模型的特征参数来实现手形的身份认证。利用轮廓点特征分布的概率密度函数来描述手形轮廓的形状变化，同时采用高斯混合模型(GMM)来估计轮廓点特征分布点的概率密度函数。

首先需要得到手形轮廓点的形心矩序列，目的是把二维手形区域的形状分析问题转化为对一维信号序列的分析问题。然后建立形心矩序列分布的高斯混合模型，高斯混合模型定义如下

$$p(r \mid \Delta) = \sum_{i=1}^{K} \alpha_i N(r, u_i, \sigma_i)$$

其中：$N(r,u_i,\sigma_i)=\frac{1}{\sqrt{2\pi\sigma_i}}\exp\left(-\frac{(r-u_i)^2}{2\sigma_i^2}\right)$。

最后根据训练数据确定高斯分量数据 K 和高斯分量参数 Δ。得到每个手指的模型参数后，即可采用 Bhattacharyya 距离量度对应手指的相似性，从而做出最后的匹配结果。

6.5 人耳识别技术

6.5.1 概述

人耳识别技术是 20 世纪 90 年代末开始兴起的一种生物特征识别技术。人耳具有独特的生理特征和观测角度优势，使人耳识别技术具有相当的理论研究价值和实际应用前景。从生理解剖学上，人的外耳分耳郭和外耳道。人耳识别的对象实际上的外耳裸露在外的耳郭，就是人们习惯上称的耳朵。人耳识别技术既可作为其他生物识别技术的有益补充，也可以单独应用于一些个体身份鉴别的场合。

人耳朵的解剖结构包括耳轮、耳垂、对耳轮、耳甲腔、耳屏、对耳屏、耳轮脚、三角窝等，如图 6-20 所示。在基于生物特征的身份鉴别技术中，人耳与人脸最相似，但是与脸相比，外耳图像尺寸更小，意味着计算量更小，且人耳不受表情、化妆的影响，不随年龄的增长而改变。所以人耳识别技术可以作为一种简单、可靠的个体识别技术。

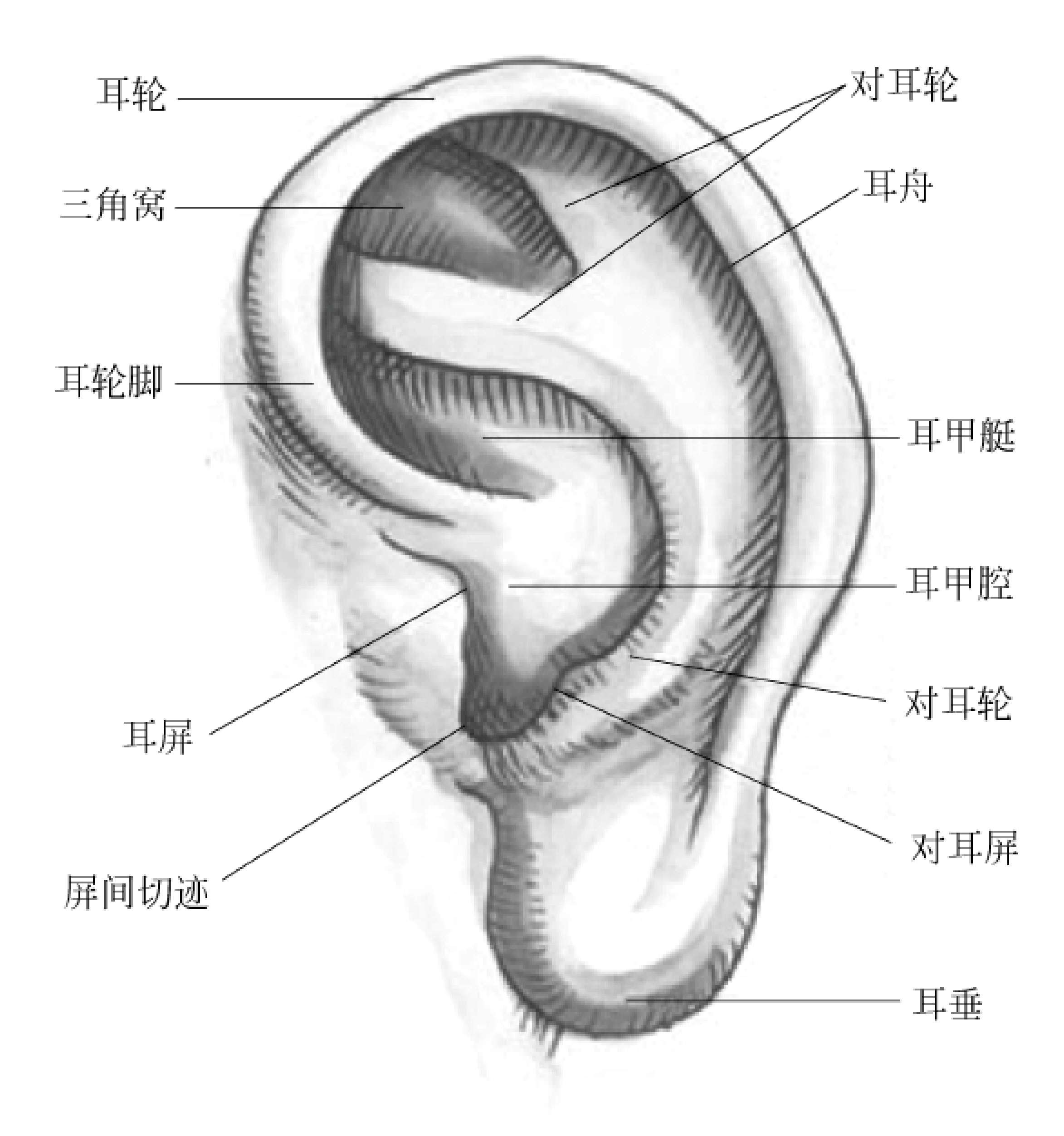


图 6-20 人耳外耳结构图

早在 1946 年美国就已经发表了关于人耳识别系统的相关论述，通过在一张放大的二维图像上放置一个有 8 根轮辐的透明罗盘，在耳朵周围确定 12 个观测点，然后将待测图像投影到特定标准画板的指定区域，最后在图像中提取测量段识别不同的人耳。一套完整的人耳自动识别系统一般包括以下几个过程：人耳图像采集、图像预处理、人耳图像的边缘检测与分割、特征提取、样本训练和模板匹配。图像的采集一般通过摄像机或 CCD 照相机采集一定数量的人耳图像，建立耳图库。预处理阶段通常包括降噪、增强以及归一化、去除噪声、进行光照补偿等处理。

6.5.2 人耳识别方法

1. 主元分析方法(PCA)

PCA 是生物特征识别研究中广泛使用的一种技术，在人脸识别领域已经进行了详细分析。PCA 是一种降维技术，它根据图像的统计特性进行正交变换，以消除原有向量各分量间的相关性，变换得到对应特征值依次递减的特征向量。具体的 PCA 算法识别过程如下。

(1) 将人耳和人脸图像进行剪裁、归一化、屏蔽非耳朵区域、补灯光等操作。

(2) 用主元分析法训练得到特征耳。

(3) 用最近邻法对测试图像的特征向量与注册库中的特征向量进行匹配。

2. 使用各种组合技术的神经网络方法

Morenno 等人使用神经网络设计了 3 种分类器，来对人耳图像进行比对和识别。

(1) 使用外耳特征点进行识别。使用双 Sobel 算子得到外耳轮廓图作为外耳特征点构成的特征向量,即神经元的输入,识别率为 43%。

(2) 使用外耳形态进行识别。使用上述技术构造大小为 $h \times v$ 像素的外耳轮廓图,然后在水平方向上进行 h 分割,在垂直方向上进行 v 分割,再在对焦方向上进行 $2(h+v)$ 分割,对人耳轮廓图中交叉点个数和不同分割构成的向量进行归一化,得到每幅图的形态学特征向量作为神经元的输入,识别率为 83%。

(3) 使用压缩网络进行识别。首先提取原始图像显著的统计特征和宏观特征,即压缩特征。这个压缩向量是原始图像的一个中间编码表示,它构成第二阶段执行识别任务的神经网络的神经源输入。

3. 力场转换方法

Hurley 等人模仿自然界的电磁力场过程,提出了一种力场转换理论。在该理论中整幅图像被转换为一个力场,该力场的形成是通过假定图像上每个像素点对其他所有像素点均施加一个等方向性的力;力与像素灰度成正比,像素间距的平方成反比。由此,得到一个与力场相关的势能面。在待检测的耳周围放置一组单位亮度的测试像素点,它们成封闭形将耳包围。每一个测试像素点在力场的拉动下朝着潜在势阱运动,直至到达平衡位置,即势阱的中心,其产生的运动轨迹形成场线。

由于在每一点的力场是唯一的,所有到达给定点的场线都会沿着同样的路径,并从该点继续向前运动从而形成"渠"。50 个测试像素点呈椭圆形被放置在力场中,测试像素点经过多次计算形成场线,并且能够从图像中观察到渠的形成过程。潜在的势阱位置被提取出来作为基本特征向量描述人耳的特征点,并证明不同的耳朵,其势能通道与势阱是唯一的。

4. 几何学方法

Michal 提出了一种几何学方法来提取特征点。它用自己提出的算法进行轮廓提取,然后进行二值化、坐标归一化,找到其质心。质心是为特征提取所找的参考点,以质心作为参考点可以使图像满足平移、旋转、大小不变性。第一个特征向量 V 由几何信息构成。以质心为圆心画 N 个不同半径的同心圆。对每一个圆,算出其与耳郭的交点数量 l 以及相邻交点间的距离 d。根据圆半径的不同,把所有交点及相应信息存入第一个特征向量 V 中。第二个特征向量 F 由特征点信息构成,即由耳郭线端点、分叉点和与圆的交点构成。对耳郭线上的每一点,找出其相邻的八个点中属于耳郭 $g=1$ 的个数 N_{8c}。$N_{8c}=1$ 则说明该点是耳郭线端点。若 $N_{8c}>2$ 则说明该点是耳郭线的交叉点。

6.6 典型应用实例

生物特征识别技术应用于社会生产中的各个领域,在人们的生活中起着重要的作用。其中一个典型实例就是生物特征识别技术在高安全级门禁管理系统中的应用。

以智能卡为载体、生物识别技术作为身份认证的方法,在高安全级门禁系统中广泛使用。每位具有权限的管理员都会拥有一个智能卡出入证,它是在普通出入证中增加智能芯片而成的,存储有该员工的姓名、性别、部门、身份证号等个人信息。更重要的是,出于安全和防伪的考虑,在智能芯片中存储有脸部图像、指纹或虹膜等生物特征信息。具有生物特征

识别的门禁系统如图 6-21 所示。

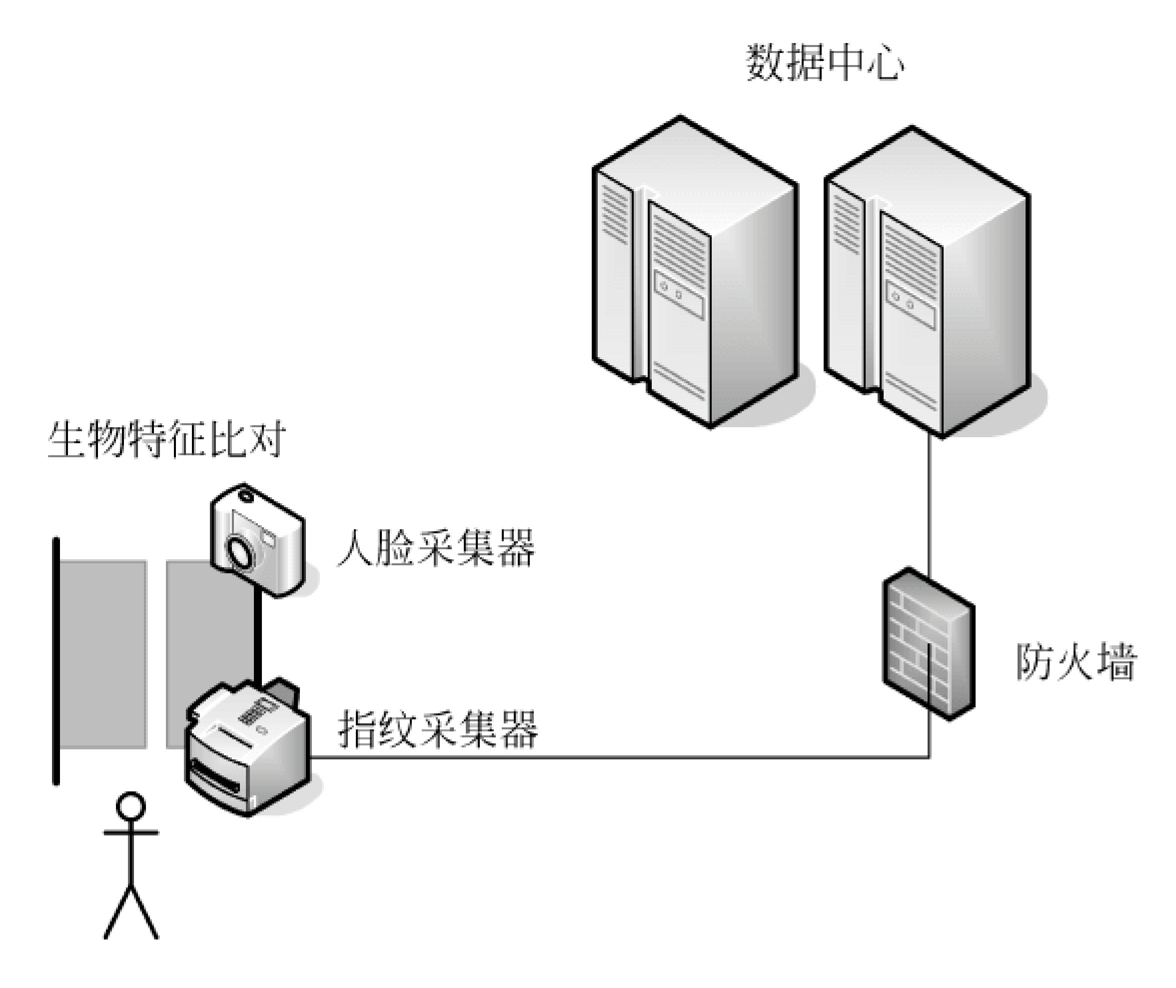


图 6-21 具有生物特征识别的门禁系统

智能卡出入证增加电子信息和生物特征识别,加强了诸如银行、保密单位等重点安全防范单位的出入管理并防止证件伪造和变造,预防抢劫等突发事件;且为人员管理提供了方便。

智能卡出入证的申办使用流程:首先,出入敏感区域的相关人员,到相关的门禁管理部门申请出入证,填写个人信息,并采集生物特征信息(包括脸部图像的采集,指纹或虹膜信息的采集);然后通过个人化操作把填写的个人信息和生物特征信息存储到智能卡出入证中;最后由门禁管理部门与人事部门核准后激活,便可正常使用。对于智能出入证丢失的情况先申请作废原卡,然后按照上述步骤进行补卡。

参考文献

[1] Nalini K. Ratha,Kalle Karu,Shao Yun Chen. A Real-Time Matching System for Large Fingerprint Databases. IEEE Trans on Pattern Analysis and Machine Intelligence,1996,8(18):342~360

[2] 杨小青,杨浩,陈意林. 自动指纹识别系统中的图像预处理技术. 重庆科技学院学报,2006,(8):91~95

[3] 夏德深,傅德胜. 现代图像处理技术与应用. 南京:东南大学出版社,1997

[4] 张翠平,苏光大. 人脸识别技术综述. 中国图像图形学报,2000,5(11):885~894

[5] 吕岳,施鹏飞. 一种实用并行细化算法及其实现. 计算机工程与设计,2000,21(4):53~56

[6] 郭晶莹,吴晴. 基于 Matlab 实现的指纹图像细节特征提取. 计算机工程与应用,2007,24(1):182~185

[7] Jain AK, Prabhakar S, Hong L. A Multichannel Approach to Fingerprint Classification. IEEE Transactions on Pattern Analysis and Machine Intelligence,1999,21(4):348~358

[8] Kalle Karu,Anil K. Jain,Fingerprint Classification. Pattern Recognition,1996,3(29):389~404

[9] Yang G Z,Huang T S. Human face detection in a complex background. Pattern Recognition,1994,27(1):53~63

[10] 卢春雨,张长水,闻方等. 基于区域特征的快速人脸检测法. 清华大学学报(自然科学版),1999,

39(1)：101～105

[11] Yow K-C，Cipolla R. Feature-based human face detection image and vision. Computing 1997，5(9)：713～735

[12] 王延江，袁宗宝，唐晚芳. 一种快速彩色图像中复杂背景下人脸检测方法. 电子学报，2002，30(10)：1566～1569

[13] Sakai T，Nagao M，Fujibayashi S. Line Extraction and Pattern Detection in a Photograph，Pattern Recognition，1969，(1)：233～248

[14] Yuille A，Hallinan P，Cohen D. Feature Extraction from Faces Using Deformable Templates. Computer Vision，1992，2(8)：99～111

[15] 张忠波. 复杂背景下的人脸的检测与识别. 吉林：吉林大学，2005

[16] Rowley H A，Baloja S kannada T. Neural network-based Bayesian framework for face detection. IEEE trans on Pattern Analysis and Machine Intelligence，1998，20(1)：23～38

[17] Rowley H，Baluja S，Kanande T. Rotation invariant neural network-based face detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition，1998：38～44

[18] Nefian A. V，Hayes M. H. Face detection and recognition using hidden markov models. Conf. on image processing Chicago，1998，1：141～145

[19] 孙宁，邹采荣，赵力. 人脸检测综述. 电路与系统学报，2006，11(6)：101～108

[20] Moghaddam B，Pentland A. Robabilistic visual learning for object representation. IEEE Trans on Pattern Analysis and Machine Intelligence，1997，19(7)：696～710

[21] Eremad K，Wilson C L，Sirohey S. Discriminant analysis for recognition of human face images. J. of Optical Society of American，1997，8(14)：1724～1733

[22] Lades M，Vorbruggen J C. Distortion invariance object recognition in the dynamic link architecture. IEEE Trans On Computer，1993，42(3)：300～311

[23] Froment J. Image comprehensive through level lines and wavelet packet Analysis. Journal of De Physique，2001

[24] Pencv P S，Atick J. Local feature analysis：a general statistical theory for object representation. Network Computation in Neural System，1996，7(3)：254～261

[25] Theodoridis S，Koutroumbas K 著，李晶皎译. 模式识别. 第 3 版. 北京：电子工业出版社，2006

[26] Elad M，Feuer A. Super-resolution reconstruction of image sequence. IEEE Trans on Pattern Analysis and Machine Intelligence，1999，21(9)：817～834

[27] Henning Biermann，Adi Levin，Denis Zorin. Piecewise smooth subdivision surface with normal control. Proceedings of the 27th annual conference on Computer graphics and interactive techniques table of contents，2000：113～120

第7章

安全规范

随着智能卡应用到越来越多的领域,其规范性与应用的安全性问题日益凸显,安全规范是其中一个重要的因素,本章将对智能卡涉及的相关安全规范作简要的介绍。

本章主要描述了:

(1) 安全操作命令集规范 ISO/IEC 7816-8。

(2) 基于整数因子分解的带有信息恢复的数字签名方案 ISO/IEC 9796-2 标准。

(3) RSA 公司制定的 PKCS 规范的简单介绍。

(4) 有限域上的椭圆曲线加密算法 ISO/IEC 15946 标准的简单介绍。

(5) GP 规范中四种安全通信通道 SCP 规范。

(6) 中国金融集成电路(IC)卡规范 PBOC 2.0 第一部分:卡片规范。

7.1 ISO/IEC 7816-8

ISO/IEC 7816-8 为安全操作的命令集规范,此规范指定了可能被应用于密码操作的互操作命令,当然并不是所有符合本规范的卡片都要支持所有的命令或者命令的所有选项。下面介绍几种主要的安全操作命令。

7.1.1 产生非对称密钥对

产生非对称密钥对命令可以一步完成,或多步完成,可以采用 ISO/IEC 7816-4 规定的命令链的方式。此命令主要执行以下的操作:

(1) 产生一组非对称密钥对(公钥和私钥),并且保存到卡片上。

(2) 或者访问之前存储在卡上的一组非对称密钥对。

为了设置产生密钥的相关参数(例如,参考算法),可以在此命令前执行"管理安全环境"命令。命令的各个参数如表 7-1 所示。

表 7-1 产生非对称密钥对命令-响应对

字段	备注(Hex)
CLA	参照 ISO/IEC 7816-4 规定
INS	46 或 47
P1	产生控制,参照表 7-2
P2	00 或者产生的密钥的参数
LC 域	Nc=0 时此项不存在,Nc>0 时存在

续表

字段	备注(Hex)
数据域	不存在；如果 P1-P2 为‘0000’则为专用数据；如果 P1-P2 不是‘0000’则为与密钥产生相关的一个或若干个控制参考模板
LE 域	Ne=0 时此项不存在，Ne>0 时存在
数据域	不存在；或者数据元素或数据对象序列的公钥；或者符合一个扩展标题列表的数据对象序列
SW1-SW2	参照 ISO/IEC 7816-4

P1 含义如表 7-2 所示。

表 7-2　P1 含义

b8	b7	b6	b5	b4	b3	b2	b1	值
0	0	0	0	0	0	0	0	不提供任何信息
1	0	0	0	0	*	*	*	提供额外信息
1	0	0	0	0	—	—	*	**密钥产生**
—	—	—	—	—	*	*	0	产生密钥对
—	—	—	—	—	*	*	1	访问已存在的公钥
1	0	0	0	0	—	*	—	**返回公钥数据的格式**
—	—	—	—	—	*	0	*	公钥数据的专用格式
—	—	—	—	—	*	1	*	符合一个扩展标题列表的公钥数据的输出格式
1	0	0	0	0	*	—	—	**输出指示符**
—	—	—	—	—	0	*	*	响应数据域的公钥数据
—	—	—	—	—	1	*	*	如果 LE 域不存在无返回数据或者如果 LE 域存在为专用数据
其他值为保留值								

如果 LE 域不存在，产生的密钥对将保存在卡中(可能会在执行此条命令时，已经获知文件的一个参数)。如果该命令为访问一组密钥对的情况，命令的数据域为空。根据 INS 的奇偶，返回数据域的公钥为一个数据元素序列(‘46’)或者一个数据对象序列(‘47’)。

7.1.2　执行安全操作命令

根据命令中 P1-P2 指定的数据对象不同，此命令执行以下的几个安全操作。

(1) 加密校验和的计算。

(2) 数字签名的计算。

(3) 哈希计算。

(4) 加密校验和的核验。

(5) 数字签名的核验。

(6) 证书的核验。

(7) 加密。

(8) 解密。

执行安全操作命令的上条命令可能是“管理安全环境”命令。例如，密钥参数和算法参数可以隐性指定或者在“管理安全环境”命令中的控制参考模板中指定。只有在安全状态满

足安全属性条件下才可以执行此命令，并且此命令执行的成功与否与上条命令是否成功息息相关。如果一个标题列表或者一个扩展标题列表存在的话，它将定义形成安全操作输入的顺序和多个数据项。该命令的各个参数如表 7-3 所示。

表 7-3 执行安全操作命令的命令-响应对

字段	备注(Hex)
CLA	参照 ISO/IEC 7816-4 规定
INS	2A
P1	标识(如果存在的话，响应数据域为数据元素)或者'00'(响应数据域不存在)；'FF'为预留
P2	标识(如果存在的话，命令数据域为数据元素)或者'00'(命令数据域不存在)；'FF'为预留
LC 域	Nc=0 时此项不存在，Nc>0 时存在
数据域	不存在或为 P2 中指定的数据对象的值
LE 域	Ne=0 时此项不存在，Ne>0 时存在
数据域	不存在或为 P1 中指定的数据对象的值
SW1-SW2	参照 ISO/IEC 7816-4

执行此命令使用表 7-4 所列的输入模板。

表 7-4 输入模板

标识	值
A0	哈希计算的输入模板(对模板进行哈希计算)
A2	加密校验和核验的输入模板(对模板进行集成)
A8	数字签名核验的输入模板(对模板进行签名)
AC	数字签名计算的输入模板(对链接值域进行签名)
AE	证书核验的输入模板(链接值域进行证明)
BC	数字签名计算的输入模板(对模板进行签名)
BE	证书核验的输入模板(对模板进行证明)

表 7-5 列出了输入模板的数据对象。

表 7-5 输入数据对象

标识	值	A0	A2	A8	AC，BC	AE，BE
80	明文值	*	*	*	*	*
8E	加密校验和		*			*
90	哈希值	*		*	*	*
92	证书					*
9C	公钥			*		*
9E	数字签名			*		*

下面将对各种安全环境管理操作进行介绍。

1. 计算加密校验和操作

加密校验和操作的参数和数据域如表 7-6 所示。

表 7-6 计算加密校验和操作的参数和数据域

字　段	备注(Hex)
P1	8E
P2	80
命令数据域	用于计算加密校验和的数据
响应数据域	加密校验和

2. 计算数字签名操作

此命令用来计算数字签名,算法可以是数字签名算法,或者哈希算法与数字签名算法的集合。对于一个数字签名的计算,签名过程中将要签名或集成的数据在命令的数据域中指定,或者在之前的命令中指定。根据输入结构的不同,P2 可以为 9A、AC 和 BC。具体情况如表 7-7 所示。

表 7-7 计算数字签名的参数和数据域

字　段	备注(Hex)
P1	9E
P2	9A, AC, BC
命令数据域	不存在(数据已经存在于卡片中)或者 如果 P2=9A,签名过程中的要签名或集成的数据,或者 如果 P2=AC,数据对象,它的值域将要在签名过程中被签名或者集成,或者 如果 P2=BC,签名过程中的要签名或集成的数据对象
响应数据域	数字签名

如果数字签名的输入中包含辅助数据,控制参考模板中应该包括参考。如果辅助数据的参考数据对象为空,那么卡片将插入辅助数据。

此命令执行结束后,卡片将返回一个数字签名(P1='9E')。

3. 哈希操作

此命令通过执行以下的操作来完成哈希值的计算:

(1) 卡内全部计算哈希。

(2) 卡内部分计算哈希(例如,哈希操作的最后一轮)。

哈希值的控制参考模板('AA''AB')标识用于哈希值计算的算法参数(参见 ISO/IEC 7816-4)。对于哈希值结果,分为以下两种情况:

(1) 卡片存储哈希值用于下一条命令,LE 则不存在。

(2) 卡片在响应中返回哈希值,LE 值必须为一定的长度,哈希操作的参数与数据域参见表 7-8。

表 7-8 哈希操作的参数和数据域

字　段	备注(Hex)
P1	90
P2	80 或 A0
命令数据域	如果 P2=80,用于哈希的数据或者 如果 P2=A0,哈希计算相关的数据对象
响应数据域	哈希值或者不存在

4. 核验加密校验和操作

核验加密校验的参数和数据域参见表7-9。

表7-9 核验加密校验和的参数和数据域

字　段	备注(Hex)
P1	0
P2	A2
命令数据域	操作相关的数据对象(例如,80,8E)
响应数据域	不存在

5. 核验数据签名操作

此命令是对在命令数据域中传递的数字签名进行核验,其他相关的数据在命令链处理中传输或者存在于卡片中,算法是数字签名算法或者哈希算法和签名算法的集合。核验数字签名操作的参数和数据域如表7-10所示。

表7-10 核验数字签名操作的参数和数据域

字　段	备注(Hex)
P1	0
P2	A8
命令数据域	操作相关的数据对象(例如,9A,AC,BC,9E)
响应数据域	不存在

6. 核验证书操作

对于卡内证书的核验,用于核验的证书的数字签名作为数据对象,包含在命令的数据域中。核验过程中使用的证书认证中心的公钥应该保存在卡片中,并且被隐性选择或使用"管理安全环境"命令参考数字签名的控制参考模板。应用的算法是隐性获取或参考数字签名的控制参考模板。如果在核验过程中使用其他的数据对象(例如,哈希值),这些数据对象将存储在卡片中或者通过命令链过程进行传输。

以下的两种情况应该区分清楚:

(1) 如果证书是自描述的(P2=‘BE’),卡将通过标识在证书内容中获取公钥。

(2) 如果证书不是自描述的(P2=‘AE’),卡将隐性获取证书中的公钥或者通过在标题列表中表述公钥内容的公钥标识显性获取公钥。

如果公钥存储在卡中,它将是接下来的核验数字签名操作的默认密钥。核验证书操作的参数和数据域如表7-11所示。

表7-11 核验证书操作的参数和数据域

字　段	备注(Hex)
P1	0
P2	92,AE或者BE
命令数据域	数据元素或者操作相关的数据对象(例如,9A,AC,BC,9E)
响应数据域	不存在

7. 加密操作

此操作加密在命令数据域中包含的数据。加密操作的参数和数据域如表 7-12 所示。

表 7-12 加密操作的参数和数据域

字　段	备注(Hex)
P1	82,84,86(密文)
P2	80(明文)
命令数据域	不存在(数据已经在卡片中)或者需要加密的数据
响应数据域	加密的数据

8. 解密操作

此操作解密在命令数据域中传输的数据。加密操作的参数和数据域如表 7-13 所示。

表 7-13 加密操作的参数和数据域

字　段	备注(Hex)
P1	80(明文)
P2	82,84,86(密文)
命令数据域	需要解密的数据
响应数据域	不存在(解密数据已经在卡片中)或者解密的数据

7.2 ISO/IEC 9796-2

ISO/IEC 9796-2 规范描述了基于整数因子分解的带有信息恢复的数字签名方案。

7.2.1 简 介

数字签名方案一般应用在实体认证、数据原点认证、认可认证和数据完整性,它应该符合以下的要求:

(1) 给定核验密钥(不是签名密钥)不能计算出信息的有效签名值。

(2) 给定一个签名值,它不能对一个信息进行签名或从签名值中恢复出签名密钥。

(3) 对签名者来说,都不能找到具有相同签名值的两个不同信息。

大多数的数字签名机制都是基于非对称密钥机制,且涉及以下三个操作:

(1) 产生密钥对的过程,每个密钥对包括私有签名密钥和相对应的公共核验密钥。

(2) 使用签名密钥的过程,被称为签名过程。

(3) 使用核验密钥的过程,被称为核验过程。

数字签名机制分为以下两种:

(1) 相同的签名密钥对相同信息的两次签名结果相同,此机制被称为非随机的(决定性的)。

(2) 给定信息和签名密钥,不同次的签名产生不同的签名结果,此机制被称为随机性的。

ISO/IEC 9796-2 制定的三个方案中的第一个和第三个都是决定性的(非随机性的),而

第二个为随机性的。

数字签名机制同时还可以分为以下两种：

(1) 整个信息必须同签名值一起存储或发送，此机制被称为具有附录功能的签名机制。

(2) 整个信息或者部分信息可以从签名值中恢复出来，此机制被称为具有信息恢复功能的签名机制。

在 ISO/IEC 9796-2 中规定的机制可以全部或部分恢复信息，可以减少存储和传输的费用。如果信息足够短，整个信息都可以包含在签名值中，且在核验过程中从签名值中恢复。否则，仅部分信息可以包含在信息中，剩余部分与签名一起存储和/或发送。ISO/IEC 9796-2 规定的机制都使用哈希函数对所有信息进行哈希运算。

7.2.2 范围

ISO/IEC 9796-2 详细说明了三个具有信息恢复功能的数字签名方案，这三个方案都是基于很难分解一个大数的数学问题，都能够恢复全部或部分信息。使用 ISO/IEC 9796 此部分的数字签名机制的用户都应该确保以下的属性：

(1) 用于签名的信息应该是任何长度的一个二进制串，也可能为空信息。

(2) 签名函数使用一个私有的签名密钥，而核验函数使用的是相应的公共核验密钥。每个签名实体应该使用并且秘密保存好与公共核验密钥相对应的私有签名密钥。每一个核验实体应该清楚签名实体的公共核验密钥。

(3) 使用在此标准中制定的签名方案，要求选择抗碰撞的哈希函数 h。签名机制和哈希函数之间应该有一个绑定，如果没有绑定，攻击对手就可能会声称使用弱哈希函数(不是实际的哈希函数)，因此伪造一个签名值。

(4) 签名核验者应该会有安全独立的方式，来判断使用了此标准中规定的三个签名方案的哪一个方案来产生签名值。如果使用的是签名方案 2 或者 3，签名核验者应该有方法来获知使用了哪种签名产生函数。例如，可以通过在协商的域参数中指定机制和签名产生函数，或者通过在签名者的公钥证书中包含一个签名方案和签名产生函数的标识符。签名产生函数还可以在同签名数据相关的算法标识符中指定。

(5) ISO/IEC 9796 规定的每个数字签名方案都有其特殊的选项，签名者采用的选项范围必须通过一种安全的独立的方式告知核验者。这些选项如下：

① 对于所有三个数字签名方案，核验者必须知道是否使用尾域选项 1 或 2；

② 对于数字签名方案 2 和 3，核验者必须清楚 Ls(salt S 的长度)。

例如，通过在“域参数”中指定选项选择或者通过在签名者的公钥证书中添加选项信息来获取上面提到的参数。

7.2.3 签名和核验过程模型

签名和核验过程模型可以应用到 ISO/IEC 9796-2 所有的三个方案中。这种类型的签名方案当应用到信息 M 时能够提供全部或者部分的信息恢复。

(1) 如果 M 足够短，M 就可以被完整包含在签名中，该信息就可以完全恢复。

(2) 如果 M 比较长，只能恢复部分的信息。在此情况下，M 被分为可恢复部分(签名值

中包含的有限长度的位字符串)和不可恢复部分(一个任意长度的八进制串同签名一起存储或传送)。

此模型分为三个部分：产生签名信息过程，核验签名过程，以及为完成签名方案规范规定的签名和核验的其他附加方面的细节。下面将对签名信息过程和核验签名过程进行介绍。

1. 对信息进行签名

(1) 概述

对一个信息 M 进行签名需要三个步骤，即信息分配、信息代表产生和密钥产生。

信息分配过程主要将信息分为两个部分：可恢复部分 M_1 和不可恢复部分 M_2(可能为空)。可恢复部分的长度要小于签名方案的容量 c，c 由签名方案和其密钥决定。可恢复部分在核验过程中从签名值中恢复出，核验者同时必须通过其他的方式(例如，通过发送或者存储在签名中的方式)获知不可恢复部分。因此如果信息十分短，整个信息都能够分配到可恢复部分，不可恢复部分则为空。

信息代表的输入为上面提到的信息的两个部分，输出一个格式化的字符串——信息代表，此字符串是签名产生的输入。在方案 2 和 3 中信息代表主要使用哈希函数 h。

签名产生的输入为信息代表和私有签名密钥，输出为签名和 $\sum$，这个过程是在公钥系统中完成的。

(2) 信息分配

签名方案和其密钥的选择决定了签名的容量 c，$c \geqslant 7$。信息分为两个部分，M_1 和 M_2。

可恢复信息长度为 c^*，$c^* \leqslant c$，$c^* \leqslant |M|$，并且 $c^* \equiv |M| (\bmod 8)$。对于签名方案 1，c^* 取 $c-\Delta$ 和 $|M|$ 的最小值，$\Delta = (c - |M|) \bmod 8$。

如果 $|M| = c^*$，则整个信息都是可恢复的，即 $M_1 = M$，并且 M_2 为空。

如果 $|M| > c^*$，M_1 应该为 M 最左边的 c^* 位，并且 M_2 为 M 剩余的部分，即 M_2 包含 $|M| - c^*$ 位。

以上两种情况下 $M = M_1 \parallel M_2$。其中 $\parallel$ 为信息串并接符号。

2. 核验签名

如果信息可以完全恢复，签名信息仅仅包括签名和 $\sum$；如果信息为部分恢复，签名信息就包括不可恢复部分 M_2^* 与签名和 $\sum$。只有在核验成功的情况下，签名才会被接受。

对一个签名和 $\sum$，以及不可恢复信息部分 M_2^*，需要三个步骤来完成核验 $\sum$ 和恢复 M^*，即签名开启，信息恢复和信息组装。

签名开启的输入为签名和 $\sum$ 与公共核验密钥，输出为一个恢复的信息代表 F^* 或返回一个标识表示核验失败，这个过程在公钥体系中完成。

信息恢复输入为恢复的信息代表 F^* 和不可恢复部分 M_2^*，输出信息为恢复部分 M_1^*，或者返回一个标识表示核验失败。

信息组装为恢复的信息重新组装成可恢复部分 M_1^* 和不可恢复部分 M_2^*(此部分可能为空)即 $M^* = M_1^* \parallel M_2^*$。

7.2.4 数字签名方案1

由于在智能卡的加密方案中如果选用ISO/IEC 9796-2规则一般都采用方案1,接下来就主要对此方案进行叙述。

1. 参数

参数1模长度:假设私有签名密钥模的长度为k位,k将决定签名容量c和信息代表的长度F。

参数2尾域选项:在方案1中尾域(作为信息代表结构的一部分)为一个或二个字节。尾域的最后半个字节应该一直等于'C'。

选项1($t=1$),尾域为一个字节,该字节为十六进制字符'BC';

选项2($t=2$),尾域由两个连续的字节组成,最右边的字节应该为'CC',左边字符为标识哈希函数的哈希标识符。

参数3容量:签名容量$c=k-L_h-8t-4$。

2. 信息代表产生

信息代表产生涉及以下两个步骤:

(1) 对信息进行哈希运算。信息$M(M=M_1 \parallel M_2)$输入到哈希函数h,输出为哈希值,即$H=h(M)$,H为L_h位。

(2) 格式化。K位字符应该由以下方式组成(从左至右):

① 两位为'01';

② 如果信息为全恢复($M=M_1$),此位设为'0',如果信息为部分恢复($|M_2|>0$),则设为'1';

③ 添加$k-L_h-|M_1|>8t-4$位'0';

④ 一位'1';

⑤ $|M_1|$位的M_1;

⑥ L_h位的哈希值H;

⑦ $8t$位的尾域T。

信息代表按照以下的步骤对上面所提到的字符串从左到右以每四个连续位为一个块(半字节)的方式进行操作:

(1) 第一个半字节保持不变。

(2) 如果第一个半字节的最右边位为'0',从第二个半字节开始,如果连续为'0',则用'B'代替,直到第一个不为'0'的半字节为止,此半字节用'B'进行异或。

(3) 后续位都保持不变。

(4) 删除上面所形成的字符串的第一位(都为'0'),形成一个位长为$k-1$的位字符串F。

3. 信息恢复

核验者在处理签名前必须要知道选用的哈希函数h,因此也就获得L_h。如果恢复的信息代表F^*($k-1$位的字符串)的最后一个字节为'BC',尾域包含一个字节;如果为'CC',则包含F^*的最后两个字节,第一个字节为使用的哈希函数的标识符。该标识符用来检查是否等于核验者使用的哈希函数,如果不相同则签名核验失败。

如果尾域或哈希函数标识符不能解析，或恢复的信息代表 F^* 的第一位为‘0’，签名和 $\sum$ 都将拒收，否则继续。

将一个位‘0’添加到字符串的最左边，形成一个 k 位的字符串。该字符串将按照以下的步骤对上面所提到的字符串从左到右以每四个连续位为一个块(半字节)的方式来进行操作：

(1) 第一个半字节保持不变。

(2) 如果第一个半字节的最右边位为‘0’，从第二个半字节开始每一个等于十六进制‘B’的后续半字节都是填装域部分，直到第一个后续不为十六进制‘B’的半字节截至，则该半字节与‘B’进行异或来恢复此半字节的初始值。

(3) 后续位都保持不变。

核验者可以判断出最后(最右)填装位的位置，因此可以计算出填装位的总数量。第一个半字节的第三位也可以用于判断签名提供的是部分还是全部恢复。在部分恢复的情况下，如果填装位数为 9 或更多(首位为 1，后跟 8 个或更多的‘0’填装位)则拒绝签名和 $\sum$，否则继续进行核验过程。

从修改过的 F^* 的左边删除掉直到填装域的末点前的所有位，从右边删除掉一个或两个字节的尾。剩余二进制字符串分为两个部分：

(1) 恢复的哈希值 H^* 包含最右边的 L_h 位。

(2) 信息 M_1^* 的恢复部分包含左边的剩余位。

恢复的信息部分 M_1^* 同不可恢复的信息部分 M_2 拼接，递交给哈希函数。如果哈希结果与 H^* 相同，即 $H^*=h(M_1^* \parallel M_2)$，接收签名值并且返回 M_1^*，否则拒绝签名值。

7.3 PKCS

公钥密码标准(public-key cryptography standards，PKCS)是由美国 RSA 实验室和工业界、学术界及政府联合开发的，目的是加速公钥密码技术的发展和增强公钥系统的互操作性，从而使密码技术更多地被公众接受，拥有更广泛的应用空间。PKCS 最早在 1991 年发表，只为满足一小部分从事研究公钥技术的工作者的需要，后来逐渐被更多的人接受作为技术参考文档。PKCS 包括了证书的申请、更新、作废和发布、扩展证书内容、数字签名、数字信封的格式、不断发展的 PKI 格式标准、算法和应用程序接口等一系列相关协议，是今天所有 PKI 实现的基础，许多标准都是参考并借鉴 PKCS 得来的，例如 ANSI X9、IEEE P1363、PKIX、SET、S/MIME、SSL/TLS 和 WAP/WTLS 等。

目前 PKCS 包含 15 个技术文档，每部分内容如下：

(1) PKCS＃1。RSA 加密标准，定义了基于 RSA 公开密钥算法的数据加密和数字签名方法，包括 RSA 密钥的生成、RSA 密钥的格式、加解密过程、签名算法等。

(2) PKCS＃2。涉及了 RSA 的消息摘要加密，这已被并入 PKCS＃1 中。

(3) PKCS＃3。Diffie-Hellman 密钥协议标准，描述了建立 Diffie-Hellman 密钥协议的方法，为什么双方在没有预先安排的情况下能够获得一个只有它们自己知道的双方共享的密钥，包括 Diffie-Hellman 参数的生成、Diffie-Hellman 密钥建立的两个阶段和对实体的认

证。这个 Diffie-Hellman 密钥可以在接下来的操作中使用,例如可以用来进行通信链路的加密。这个标准已经被应用于许多的安全通信中,例如 OSI 传输层和网络层协议。

(4) PKCS＃4。最初是规定 RSA 密钥语法的,现已经被包含进 PKCS＃1 中。

(5) PKCS＃5。基于口令的加密标准,描述了如何从口令中分散密钥,来加密敏感信息,例如从一个计算机到另一个计算机的私钥加密传输。

(6) PKCS＃6。扩展证书语法标准,定义了提供附加实体信息的 X.509 扩展证书属性的语法。扩展证书包含 X.509 公钥证书和一组属性,目的是为扩展原有证书的只用一个公钥就可以验证特定实体的信息的认证方法,例如增强型保密邮件(privacy-enhanced mail,PEM)。

(7) PKCS＃7。密码消息语法标准,为使用密码算法的数据规定了通用语法格式,例如数字签名和数字信封,并允许递归定义(例如一个信封可以嵌套在另一个信封中),或其他任意的属性(例如签名时间等)。目前本协议为数据提供了六种格式,包括明文数据格式、已签名数据格式、已封装数据格式、既被签名也被封装数据格式、摘要数据格式和加密数据格式。

(8) PKCS＃8。私钥信息语法标准,定义了私钥信息语法和加密私钥语法(用 PKCS＃5 中定义的基于口令的加密算法对私钥加密)。私钥信息包含了公开密钥算法的私钥和一组属性,这组属性可以为用户提供了一种与信息建立信任的简单方法。

(9) PKCS＃9。类对象和属性类型,定义了两个新的类对象和一些新的属性类型,在 PKCS＃6、PKCS＃7、PKCS＃10、PKCS＃12 和 PKCS＃15 中使用。

(10) PKCS＃10。证书请求语法规范,定义了证书请求的语法格式和证书请求的过程。证书请求包括三部分:证书请求信息、签名算法标识符和证书请求信息的数字签名。证书请求信息包含了实体的一个唯一识别名、公钥和可选的一组属性,它们一起被请求证书的实体签名,然后和证书请求信息及签名算法标识符一起发送给发证权威机构。发证权威机构首先验证证书请求是否有效,然后根据证书请求信息制作 X.509 证书。

(11) PKCS＃11。密码令牌接口标准,该标准从 1995 年发布 1.0 版本以来,经历了 V2.01、V2.10、V2.11 到 V2.20,提供了一个能完成密码信息存取、执行密码操作的被称为 Cryptoki(cryptographic token interface,密码令牌接口,是为密码令牌设备(例如智能卡)与应用程序互操作提供的接口)的 API 接口规范,采用基于对象机制来实现技术无关(任何类型的密码设备)和资源共享(多个应用访问设备),呈递给应用的是一个通用的被称为密码令牌的设备逻辑视图。Cryptoki 把应用和密码设备的详细实现细节隔离,使应用在不同的设备或运行环境中使用相同的接口,并使不同厂商的密码设备在该标准下实现互操作。标准中定义了许多密码类型和机制,并随着密码技术的发展,在不改变已定义的通用界面的前提下新的密码类型和机制可以随后加入到标准中来,而附加机制可以不间断地发布新的文档。

在 PKCS＃11 中,加密设备被抽象为一个令牌,而令牌都存在于一个槽中。应用与槽连接,从而与设备通信。槽是一个逻辑概念,在实现中可以有多种形式,例如,智能卡的读卡器可以理解成槽。在 PKCS＃11 中有很多内容,通常被称为对象,其中包括数据对象、密钥对象和证书对象等。而密钥对象又分为公钥对象、私钥对象和对称密钥对象等。PKCS＃11 的对象还可以根据其生存期和可见性划分为“令牌对象”和“会话对象”。令牌对象是一直存在的,而且每个应用程序都可以访问,不会因为会话关闭或者令牌从槽中移出而消失。而某个会话对象是一种临时对象,当会话关闭时,所有这个会话创建的对象都会消失,而且一个应用程序不能访问另一个应用程序创建的会话对象。根据对象访问权限要求,可以把

对象分为“私有对象”和“公有对象”，公有对象不需要授权就可以访问，而私有对象必须通过令牌的授权才能够访问。

PKCS＃11 将使用设备的用户分为两个，一个是安全管理员用户，一个是普通用户。安全管理员用户登录后可以初始化令牌、设置普通用户的个人 PIN 码和操作一些公有对象，普通用户登录后可以访问令牌的所有对象，包括私有对象和公有对象，修改自己的 PIN，可以在登录后调用设备完成各种密码相关操作。PKCS＃11 规定，应用程序必须通过会话来访问令牌，会话提供了一个从应用程序到令牌的逻辑连接，整个 PKCS＃11 的权限控制体系结构主要是通过不同的会话状态来体现的，不同的会话状态被赋予不同的权限。

图 7-1 为 PKCS＃11 的实现结构图。

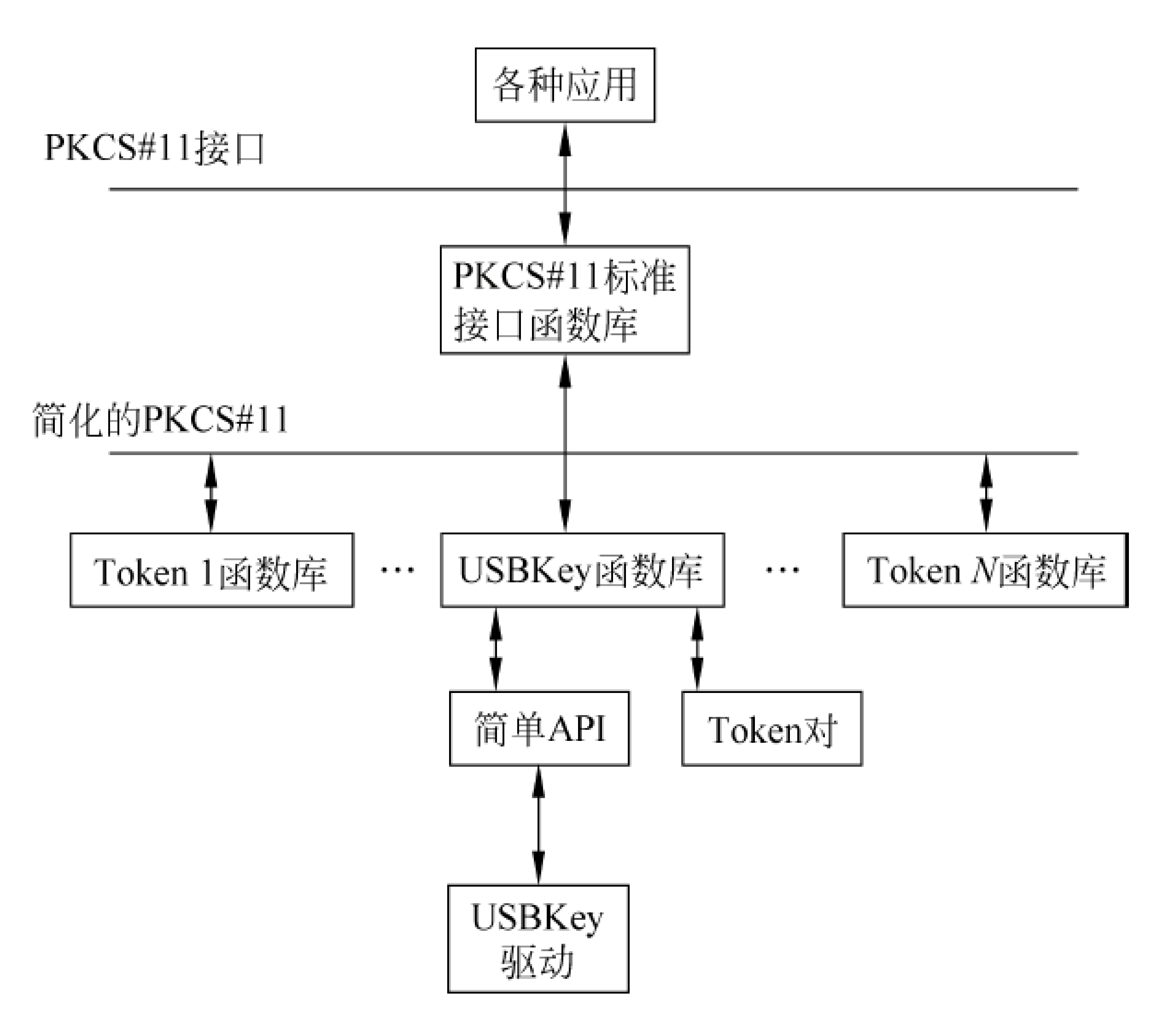


图 7-1　PKCS＃11 的实现结构图

(12) PKCS＃12。个人信息交换语法标准，定义了个人身份信息（包括私钥、证书、各种秘密和扩展字段）的语法格式。主要为了传输、备份、恢复数字证书和它们相关的在公钥加密系统里的公钥或私钥，用于输出数字证书给其他计算机、移动存储设备或智能卡。

PKCS＃12 定义了四种私有模式与完整模式的组合。私有模式使用加密算法保护个人信息不被窃取。完整模式保护个人信息不被篡改。私有模式分为如下两类：

- 公钥私有模式：在个人信息的颁发端，颁发者使用接收者的加密公钥加密个人信息。接收者通过对应的私钥就能解密出个人信息。
- 密码私有模式：在个人信息的颁发端，颁发者使用从用户名以及一个私有口令中导出的密钥，利用对称加密算法加密个人信息。

完整模式分为如下两类：

- 公钥完整性模式：该模式的完整性由使用个人信息的颁发端，由颁发者的私有密钥对整个 PDU(protocol data unit)进行签名保障。签名可以通过对应的公钥验证。
- 密码完整性模式：该模式的完整性由从秘密的完整口令中导出的消息验证码保障。

(13) PDCS＃13。椭圆曲线密码标准，目前正在开发完善中。它包括椭圆曲线参数的生成和验证、密钥生成和验证、数字签名和公钥加密，还有密钥协定，以及参数、密钥和方案

标识的 ASN.1 语法。

(14) PKCS＃14。伪随机数产生标准，目前正在开发完善中。PKI 中用到的许多基本的密码学函数，如密钥生成和 Diffie-Hellman 共享密钥协议，都需要使用随机数。然而，如果“随机数”不是随机的，而是取自一个可预测的取值集合，那么密码学函数就不再是绝对安全了，因为它的取值被局限于一个缩小的值域中。因此，安全伪随机数的生成对于 PKI 的安全极为关键。

(15) PKCS＃15。密码令牌信息语法标准。为了增强密码令牌与应用环境的互操作性，本规范通过定义存储在密码令牌中与安全相关的文件(如密钥、证书、认证对象和数据对象)和目录的通用格式，使应用环境只需提供符合标准的 PKCS＃15 解释器，即可实现密码令牌的平台无关、厂商无关和应用无关的特性。在密码令牌与应用环境的互操作中，PKCS＃15 起到了中间解释层的作用，扮演翻译家的角色，实现了卡的内格式与应用程序支持的数据格式的转换。

7.4 ISO/IEC 15946

有限域上的椭圆曲线加密体系是一个可以替代 RSA 算法的算法。基于公钥加密体系的椭圆曲线要符合以下的规则：

(1) 每个椭圆曲线都具有二进制“＋”操作形成一个有限的阿贝尔群。

(2) 椭圆曲线上的群规则可以扩展到点群上的离散幂操作。

(3) 基于椭圆曲线的离散幂运算，可以衍生出 Diffie-Hellman 和 ElGamal 公钥方案的椭圆曲线。

此公钥系统的安全性主要在于决定椭圆曲线点群上的离散对数的数学难题。以目前的知识而论，决定离散对数的数学问题比大整数的因式分解和在有限域上的离散对数计算要难得多。自从 1985 年 Miller 和 Koblitz 各自建议使用椭圆曲线的公钥加密系统以来，真正成功攻击过椭圆曲线离散对数的行为还没有被报道过。相对于 RSA 系统或者使用某些有限域的乘法群的传统离散对数来说，基于椭圆曲线的公钥系统将使用更短的参数，将产生更短的数字签名和系统参数以及完全避免了额外的大整数的算术运算。

ISO/IEC 15946 描述了基于椭圆曲线的公钥加密技术，包括秘密密钥系统的密钥建立和数字签名机制，以及基于椭圆曲线的具有信息恢复系统的数字签名机制。

ISO/IEC 15946 分成 4 个部分：

(1) ISO/IEC 15946-1：一般信息；

(2) ISO/IEC 15946-2：数字签名；

(3) ISO/IEC 15946-3：密钥建立；

(4) ISO/IEC 15946-4：带消息恢复的数字签名。

7.5 Global Platform SCP

Global Platform(以下简称 GP)联盟是一个由支付与商业领域的大公司、政府部门以及销售商主导的一个组织。GP 联盟是第一个制定跨不同行业的智能卡规范的组织，目前该

联盟拥有包括金融机构、电信运营商、智能卡和终端制造商以及软件开发公司在内的50家成员单位。该联盟由会员直接管理,并被划分为不同的委员会(即卡、终端、系统架构和商业开发等)。各个委员会负责 Global Platform 的开发和促进工作。

GP 为卡的发行商定义了一个灵活并强大的规范(更为详细的内容可参见附录 A),目标在于为智能卡在不同和行业的应用提供一种通用的体系框架,从而为跨行业的多应用的智能卡的发展扫清障碍。由 GP 制定的智能卡规范是一种与硬件无关、与厂家无关、应用程序相互独立的智能卡的管理规范。这个规范提出了一种通用的安全性和卡的管理体系结构,从而保护智能卡底层的重要数据。

GP 规范中定义了严格的安全通信协议(secure channel protocol,SCP)来保证卡外系统和卡上应用程序之间通信的安全性。对于应用程序的动态下载和删除,都必须通过相应的完整性验证。另外,基于运行环境的应用程序隔离,也保证了应用程序可以实现自己的安全机制。最重要的是,GP 定义了一个完整的安全架构,明确了每个模块所需要承担的责任。

安全通道协议为卡内外实体在一次完整的应用会话过程中提供安全的通信信道,安全通道的建立可以分为3个顺序阶段:

(1) 安全通道的发起。此时的卡和卡外实体已经交换了所需的信息,可以进行相应的密码学运算。

(2) 安全通道的运行。经过卡和卡外实体的交互,二者可以根据会话密钥建立安全通道。

(3) 安全通道的终止。在安全通道运行的任何阶段,如果卡或卡外实体认为接收到的消息和预期的不匹配,或者消息没有包含任何期待的加密保护信息,安全通道都会终止。

目前 GP 规范中定义了四种 SCP 安全通信规范,分别标志为 SCP01、SCP02、SCP03、SCP10。其中 SCP01、SCP02 属于基于 DES 对称算法的安全通道协议,SCP03 是 2009 年 2 月份刚提出的基于 AES 对称算法的安全通道协议。SCP10 是基于非对称算法和 PKI 架构的安全通道协议。其中 SCP01 和 SCP02 用得非常广泛,本书重点介绍这两种安全通道协议。

7.5.1 GP 安全机制

GP 规范中采用多种方式的安全机制:

(1) 密钥的动态分散。安全通道中使用的密钥都是临时生成的会话密钥,安全域拥有的静态密钥仅仅用于会话密钥的生成阶段,在后面的通信过程中不再继续使用,这样就保证了安全域密钥集使用的安全性。安全域密钥集包括多个密钥,可分别用于生成加密和 MAC 的会话密钥。

(2) 双向的身份认证。卡外实体和卡内安全域分别提供了一个 8 字节的挑战值,通过共同协商进行会话密钥的生成,这样就保证了认证的公平性。通过卡外实体的签名和卡内安全域的签名,从而实现双向的身份认证。

(3) 数据的加密。在安全通道中,不仅在应用程序的下载过程中可以实现数据的加密,在应用程序使用过程中也可以进行加密操作。当卡外实体向应用程序发送某些重要信息时,这种加密机制更加重要。如法定证件应用。

(4) 不可否认性。对于下载的应用程序,有些安全域会使用私钥对其进行数字签名,

GP 用户拥有了对其不可否认性的证据。

(5) 下载的同步和完整性。多条下载命令中,规定第一条数据下载命令的序号是 0,后面的需要依次递加。如果卡收到的下载命令不符合该规定,则返回警告信息。整个应用正确下载后,则通过各种数据的完整性验证,才能允许应用程序的安装。这样可应对下载过程中的断电、文件不全等特殊情况。

(6)开放性。GP 本身的目的就是要建立一个开放性的多应用平台,使得各个厂商开发的应用可以在同一智能卡上运行。卡内的安全域就是服务提供者的代表,而且安全域本身也可以作为普通的应用程序,由发卡商的安全域负责下载和安装。

7.5.2 GP 初始化命令

符合 GP 规范的卡片,初始化和认证流程如图 7-2 所示。本节将对该流程中使用到的主要命令进行介绍。

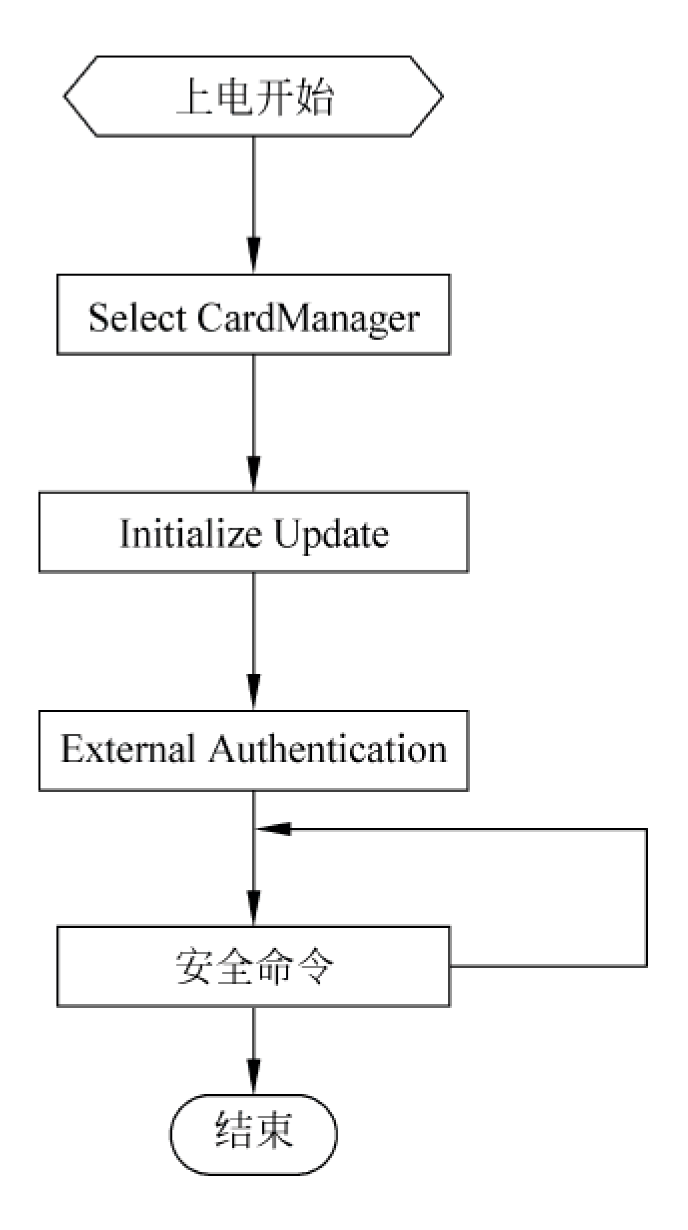


图 7-2 GP 规范的卡片初始化和认证流程图

1. Select CardManager 命令

Select 指令用于选择 CardManager 应用。命令格式如表 7-14 所示。

表 7-14 Select CardManager 命令格式

Code	Value	Meaning
CLA	00H	
INS	A4H	选择命令
P1	04H	
P2	00H	
LC	XX	08
Data	AID	A0 00 00 00 03 00 00 00(AID)

Select 指令返回选中应用的符合 ISO/IEC 7816-4 标准的文件控制标识符 FCI。

2. Initialize Update 命令

Initialize Update 指令发起一个卡片和主机之间的通信安全通道。命令格式参见表 7-15。

表 7-15　Initialize Update 命令格式

Code	Value	Meaning
CLA	80	
INS	50	Initialize Update
P1	××	密钥版本号
P2	00	固定为 00
LC	08	主机随机数长度
Data	××…××	主机随机数
LE	00	

命令响应格式参见表 7-16。

表 7-16　Initialize Update 命令响应格式

Response	密钥分散数据	10 字节
	密钥信息(密钥版本+SCP 序号)	2 字节
	卡片挑战值	8 字节
	SCP01：卡片随机数	8 字节
	SCP02：顺序计数器	2 字节
	卡片随机数	6 字节
	卡片认证密文	8 字节

3. External Authentication 命令

External Authentication 指令用于卡片认证主机,并确定安全通道使用的安全等级。命令格式及响应格式如表 7-17 所示。

表 7-17　External Authentication 命令和响应格式

Code	Value	Meaning
CLA	84～87 E0～EF	
INS	82	EXTERNAL AUTHENTICATION
P1	××	安全级别 Security Level
P2	00	固定为 00
LC	10	主机认证数据和 MAC 的长度
Data	××…××	主机认证数据和 MAC
LE		不存在
Response	SW1～SW2	

7.5.3 SCP01 协议

1. 会话密钥分散

GP 安全通道协议中规定了三种密钥，如表 7-18 所示。

表 7-18 三种密钥定义

标识	内　　容	长度
S_ENC	安全通道认证和加密	2Key
S_MAC	安全通道 MAC 码校验	2Key
DEK	敏感数据加密	2Key

其中 S_ENC 和 S_MAC 会话密钥分散流程如图 7-3 所示。

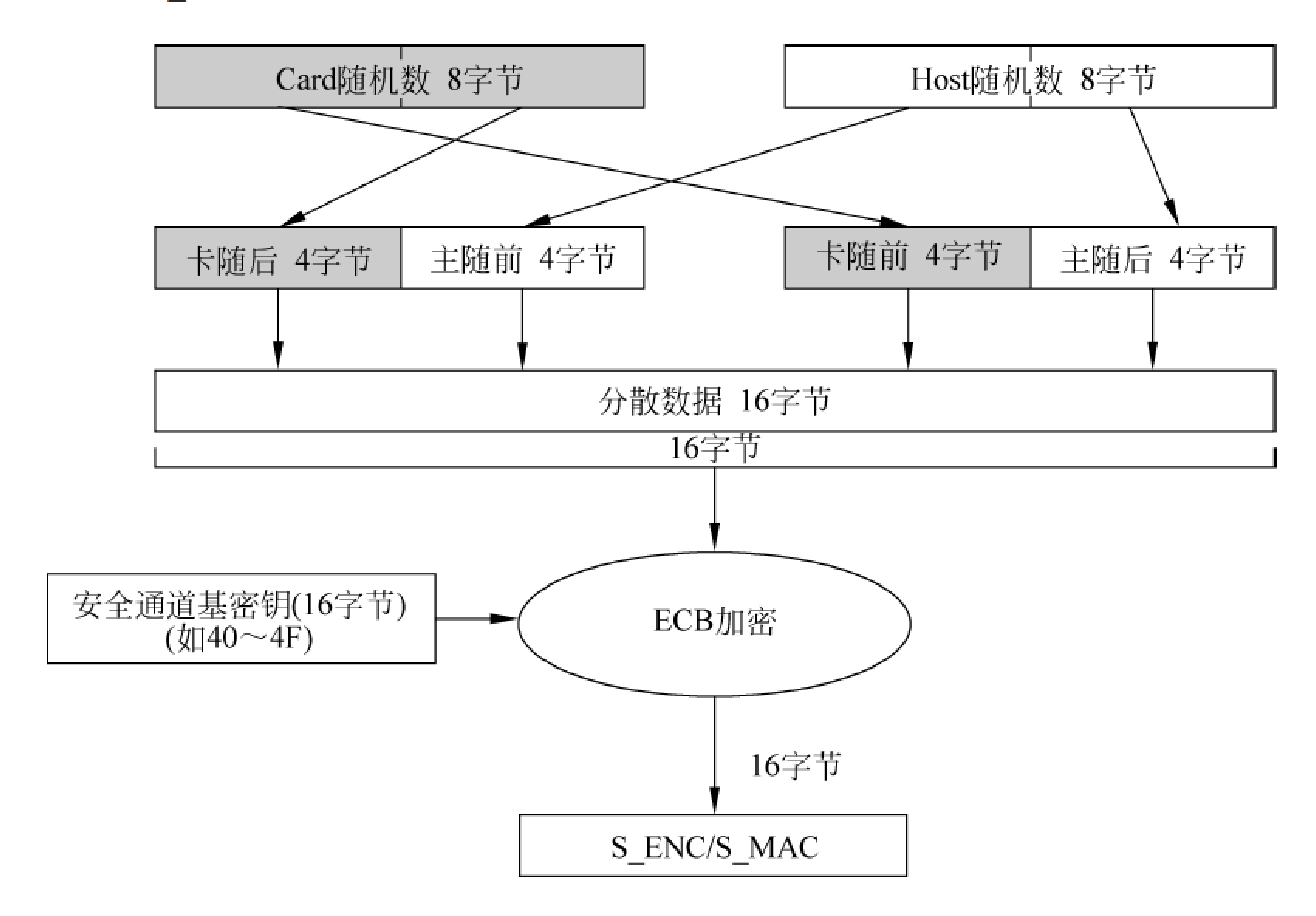


图 7-3 SCP01 会话密钥分散流程

2. 认证数据计算

卡片一侧的认证数据计算流程如图 7-4 所示。其中的 FULL 3DES MAC 算法和计算过程可参见本书第 4 章内容。

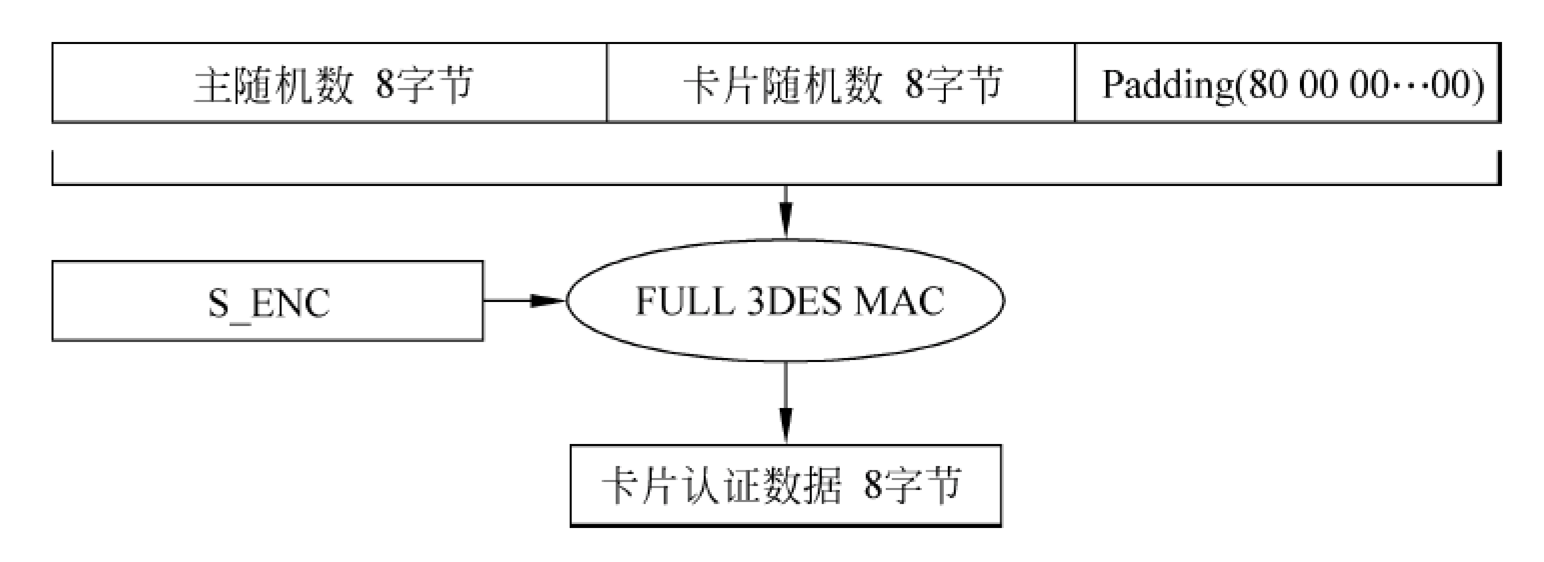


图 7-4 SCP01 卡片认证数据计算流程

主机一侧的认证数据计算流程如图 7-5 所示。其中的 FULL 3DES MAC 算法和计算过程可参见本书第 4 章内容。

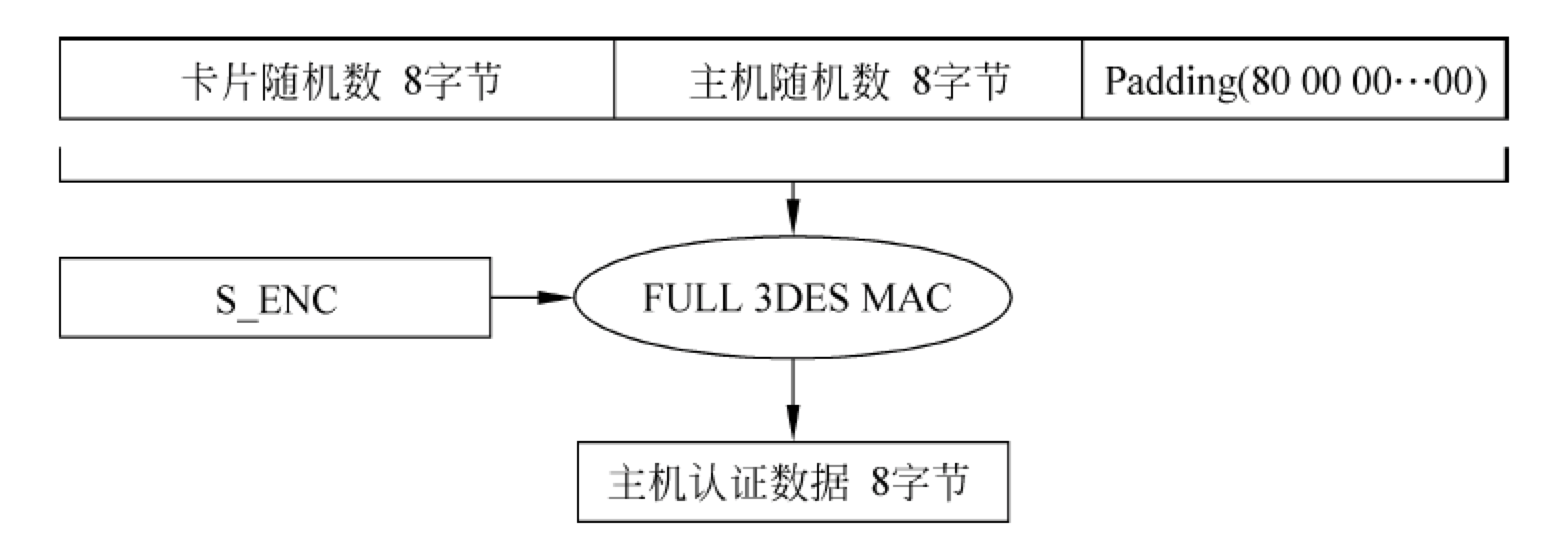


图 7-5 SCP01 主机认证数据计算流程

3. SCP01 外部认证

外部认证命令中需要计算 MAC 值,计算过程参见图 7-6 所示。其中的 FULL 3DES MAC 算法和计算过程也可参见本书第 4 章内容。

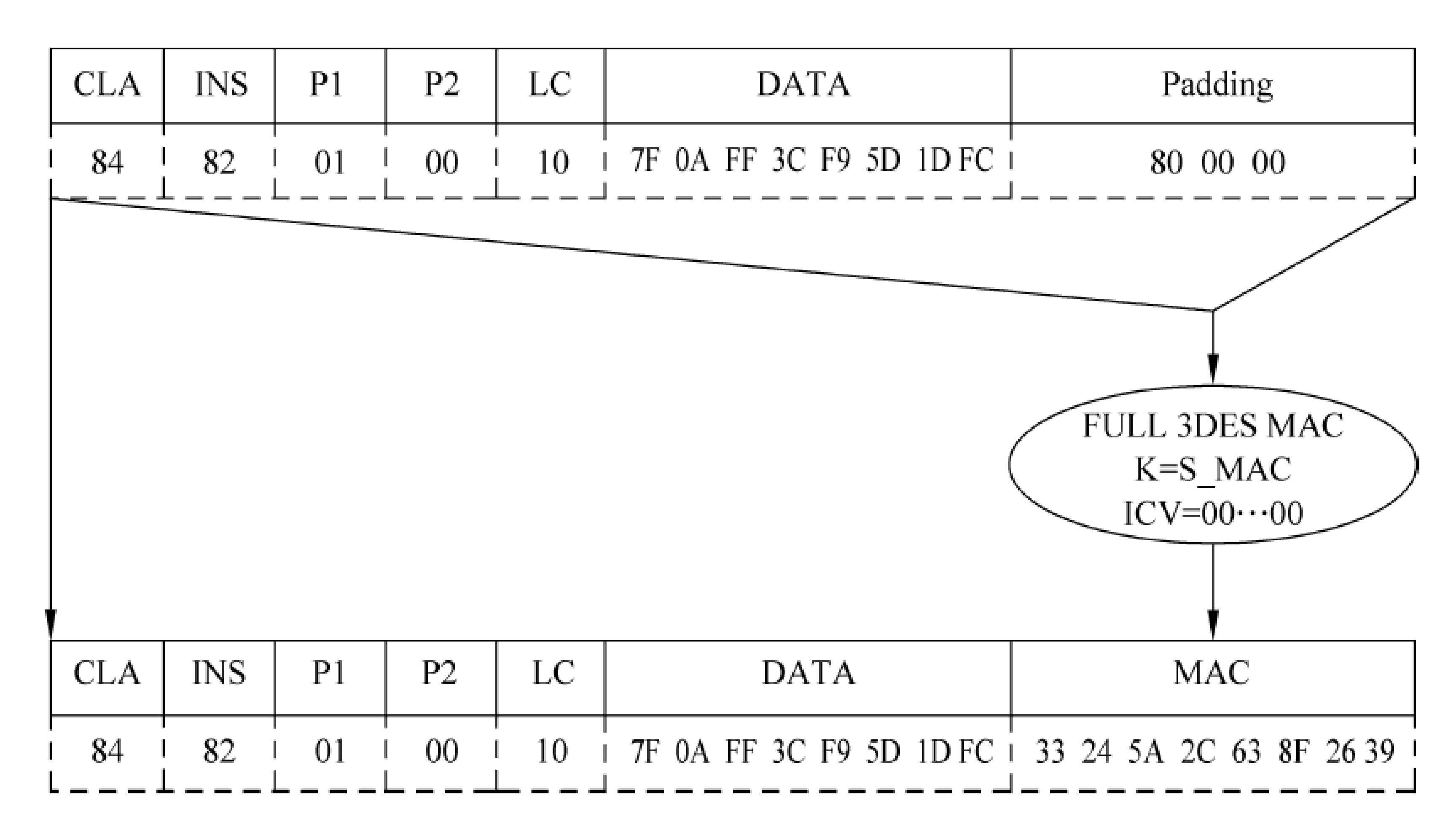


图 7-6 SCP01 外部认证命令 MAC 计算流程

4. 符合 SCP01 的 JavaCard 初始化流程

以下是符合 SCP01 的 JavaCard 初始化脚本及其运行结果,运行环境为 JCOP Shell。

```
//////////////////////////////////
cm> /atr
ATR:
3B 8B 80 01 4A 43 4F 50 34 31 37 32 47 44 54 4B
 -------------------------
Execute Time: 63ms
cm> /send 00A4040008A00000000300000000
 => 00 A4 04 00 08 A0 00 00 00 03 00 00 00 00
<= 6F 10 84 08 A0 00 00 00 03 00 00 00 A5 04 9F 65 01 FF 90 00
Status: No Error
Execute Time: 0ms
cm > set - key 255/1/DES - ECB/404142434445464748494a4b4c4d4e4f 255/2/DES - ECB/
```

```
404142434445464748494a4b4c4d4e4f 255/3/DES－ECB/404142434445464748494a4b4c4d4e4f
Execute Time: 0ms
cm> init－update 255
=> 80 50 00 00 08 95 56 04 59 A7 8B 68 5F
<= 00 00 60 18 00 22 62 90 88 35 FF 01 57 B5 9F 07 48 8E A1 98 DB 02 C8 B5 22 D3 01 96 90 00
Status: No Error

S_ENC= 30 42 D3 9F F9 89 82 A4 81 71 81 66 E9 83 0C 5F
S_MAC= 30 42 D3 9F F9 89 82 A4 81 71 81 66 E9 83 0C 5F
Execute Time: 32ms
cm> ext－auth plain
=> 84 82 00 00 10 B3 73 EA F4 19 71 24 FE 2D 18 95 3A 4D 09 B3 DD
<= 90 00
Status: No Error
Execute Time: 31ms
////////////////////////////////////
```

7.5.4 SCP02 协议

1. 会话密钥分散

SCP02 安全通道协议中规定了三种密钥，定义与 SCP01 相同。其中 S_ENC 和 S_MAC 会话密钥分散流程如图 7-7 所示。

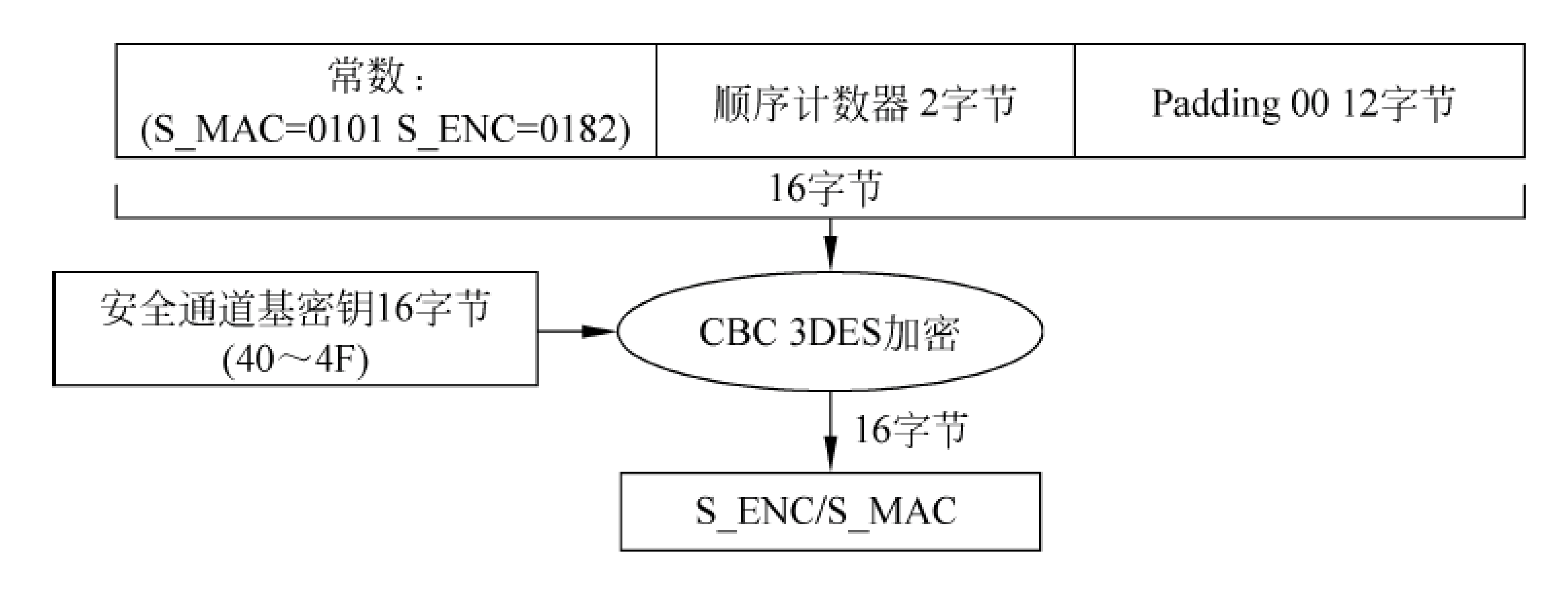


图 7-7 SCP02 会话密钥分散流程

2. 认证数据计算

卡片一侧的认证数据计算流程如图 7-8 所示。其中的 FULL 3DES MAC 算法和计算过程可参见本书第 4 章内容。

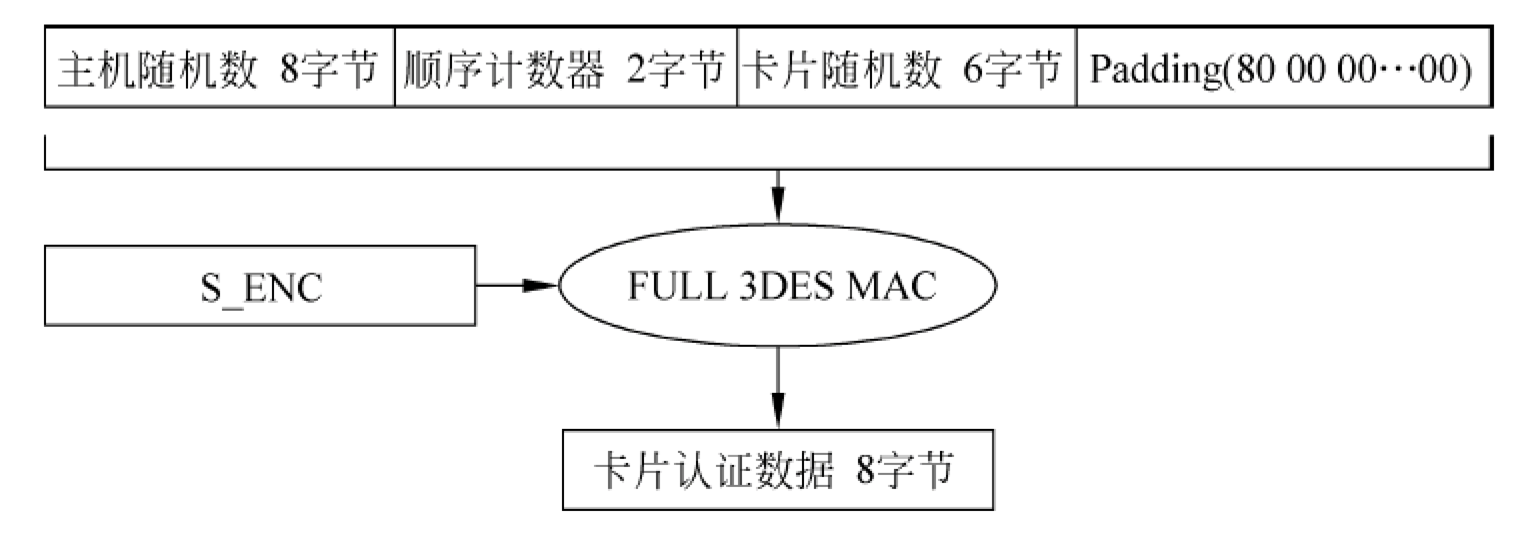


图 7-8 SCP02 卡片认证数据计算流程

主机一侧的认证数据计算流程如图 7-9 所示。其中的 FULL 3DES MAC 算法和计算过程可参见本书第 4 章内容。

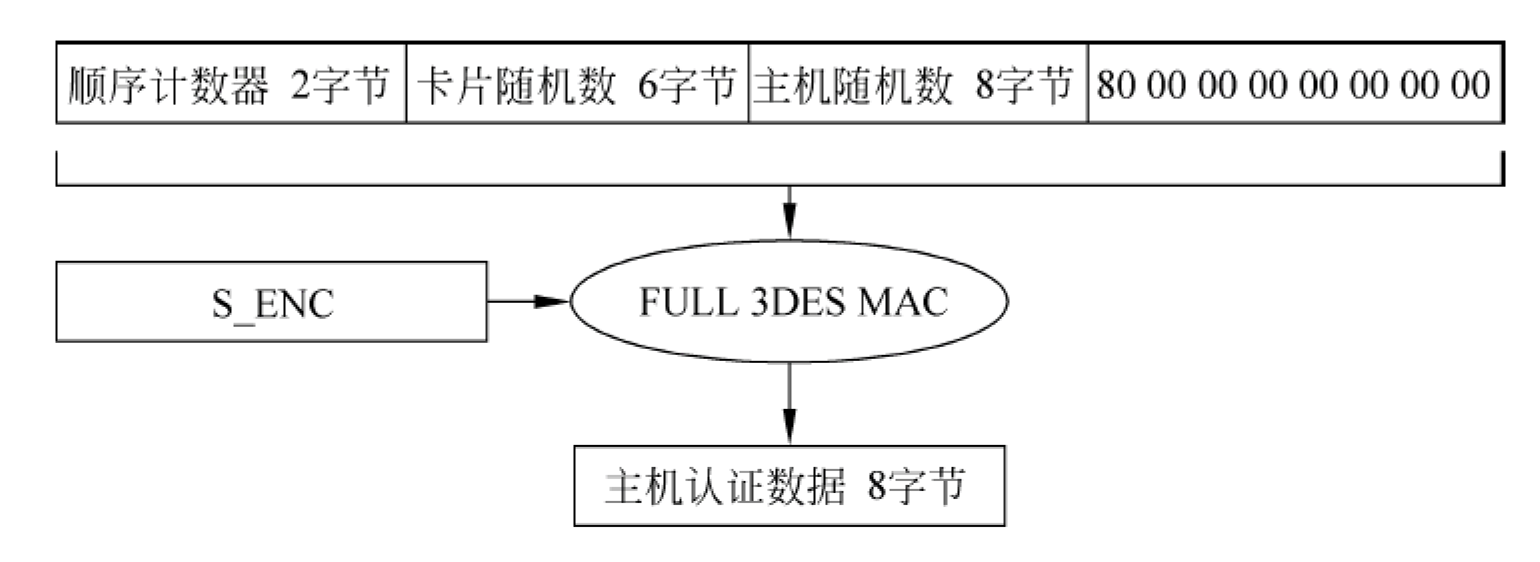


图 7-9 SCP02 主机认证数据计算流程

3. 外部认证过程

外部认证命令中需要计算 MAC 值，计算过程参见图 7-10 所示。其中的 FULL 3DES MAC 算法和计算过程也可参见本书第 4 章内容。

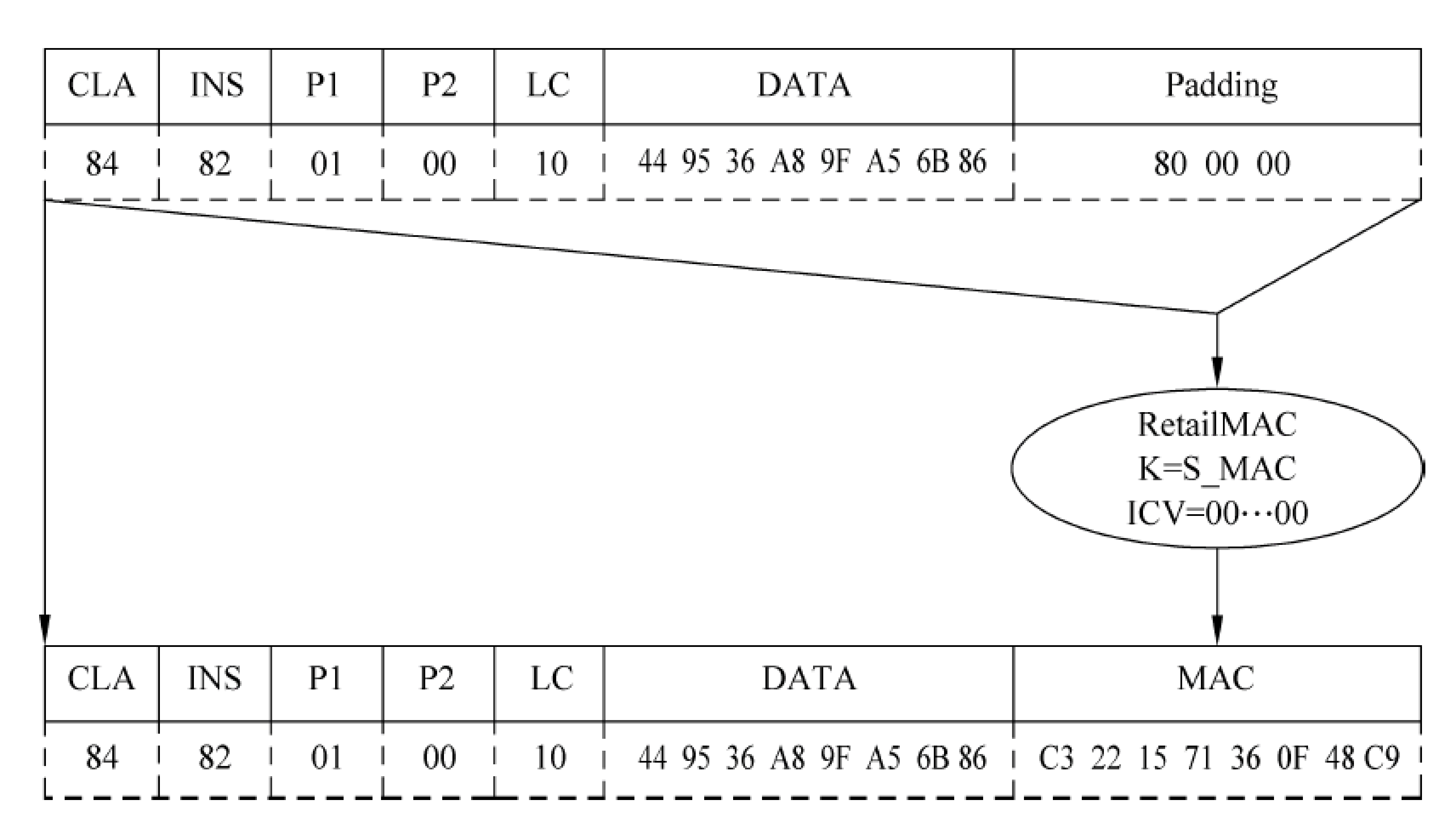


图 7-10 SCP02 外部认证命令 MAC 计算流程

4. 符合 SCP02 的 JavaCard 初始化流程

以下是符合 SCP02 的 JavaCard 初始化脚本及其运行结果，运行环境为 JCOP Shell。

```
/////////////////////////////////////////////
cm> /atr
ATR:
3B 88 80 01 4A 43 4F 50 76 32 34 31 5E
 -------------------------
Execute Time: 78ms
cm> /send 00A4040008A000000003000000000
 => 00 A4 04 00 08 A0 00 00 00 03 00 00 00 00
<=  6F 65 84 08 A0 00 00 00 03 00 00 00 A5 59 9F 65 01 FF 9F 6E 06 47 91 81 02 31 00 73 4A 06 07 2A
86 48 86 FC 6B 01 60 0C 06 0A 2A 86 48 86 FC 6B 02 02 01 01 63 09 06 07 2A 86 48 86 FC 6B 03 64 0B
06 09 2A 86 48 86 FC 6B 04 02 15 65 0B 06 09 2B 85 10 86 48 64 02 01 03 66 0C 06 0A 2B 06 01 04 01
2A 02 6E 01 02 90 00
```

```
Status: No Error
Execute Time: 15ms
cm > set - key 255/1/DES - ECB/404142434445464748494a4b4c4d4e4f 255/2/DES - ECB/
404142434445464748494a4b4c4d4e4f 255/3/DES - ECB/404142434445464748494a4b4c4d4e4f
Execute Time: 0ms
cm > init - update 255
=> 80 50 00 00 08 62 98 71 E9 2E 46 75 F3
<= 00 00 81 96 00 10 10 91 39 76 FF 02 00 B8 9B 3F FA C3 37 BE 11 2D 15 F8 CA 59 4B 7B 90 00
Status: No Error

S_ENC = 62 AE 46 47 AC 3C 33 9C 68 5D 61 9F 67 D6 73 FA
S_MAC = C5 21 3F D3 B3 F9 0F E6 A6 2D 2E CA 0C A5 F1 10
Execute Time: 47ms
cm > ext - auth plain
=> 84 82 00 00 10 47 8F A2 D5 1E 46 46 07 57 7E 6B 73 53 B6 BF 9D
<= 90 00
Status: No Error
Execute Time: 47ms
////////////////////////////////////////////////
```

7.6 PBOC 2.0 规范

在国际 EMV 迁移和国内银行卡产业蓬勃发展的大环境下，在中国人民银行的积极倡导下，我国也启动了金融卡芯片化迁移的计划。为了降低我国外卡收单的风险损失、降低伪卡欺诈率、扩大银行卡应用范围、使国民能更加方便安全地使用银行卡，中国人民银行参考国外先进经验，结合国内具体情况，制定了积极应对、审慎实施的指导方针和“先标准、后试点，先手单、后发卡，先外卡、后内卡”的实施策略。2005 年 3 月 13 日，中国人民银行发布第 55 号文，正式颁发了《中国金融集成电路(IC)卡规范》(简称 PBOC 2.0)。该规范补充完善电子钱包/存折应用；增加借/贷记应用；增加非接触式 IC 卡物理特性标准；增加电子钱包扩展应用指南、借/贷记应用个人化指南等内容。

因为本书的描述对象为智能卡，所以本节着重介绍《中国金融集成电路(IC)卡电子钱包/电子存折规范的第一部分：卡片规范》，该规范适用于由银行发行或接受的金融 IC 卡，其使用对象主要是与金融 IC 卡应用相关的卡片设计、制造、管理、发行、受理以及应用系统的研制、开发、集成和维护等部门(单位)，也可以作为其他行业 IC 卡应用的参考。卡片规范中的安全机制定义了金融应用中有关安全的总体要求、加密算法和安全机制。

7.6.1 基本安全要求

1. 共存应用

为了独立地管理一张卡上不同应用之间的安全问题，每一个应用应该放在一个单独的 ADF 中。即应该在应用之间设计一道“防火墙”以防止跨过应用进行非法访问。另外，每一个应用也不应该与个人化要求和卡中共存的其他应用规则发生冲突。

2. 密钥的独立性

用于一种特定功能(如扣款)的加密/解密密钥不能被任何其他功能所使用，包括保存在

IC 卡中的密钥和用来产生、派生、传输这些密钥的密钥。

7.6.2 密钥和个人密码的存放

IC 卡应该能够保证用于 RSA 算法的非对称私有密钥或用于 DES 算法的对称加密密钥在没有授权的情况下，不会被泄露出来。

如果使用个人密码，则应保证其在 IC 卡中的安全存放，且在任何情况下都不会被泄露。

7.6.3 安全报文传送

安全报文传送的目的是保证数据的可靠性、完整性和对发送方的认证。数据完整性和对发送方的认证通过使用 MAC 来实现。数据的可靠性通过对数据域的加密来得到保证。

1. 安全报文传送格式

PBOC 2.0 规范中定义的安全报文传送格式符合 ISO/IEC 7816-4 的规定。当 CLA 字节的第二个半字节等于十六进制数字'4'时，表明对发送方命令数据要采用安全报文传送。卡中的 FCI 表明某个命令的数据域的数据是否需要加密传输，是否应该以加密的方式处理，如表 7-19 所示。

表 7-19 安全报文标识位

b4	b3	b2	b1	说明
0	0	×	×	不需要安全报文
0	1	×	×	需要安全报文

2. 报文完整性和验证

MAC 是使用命令的所有元素(包括命令头)产生的。一条命令的完整性，包括命令数据域(如果存在的话)中的数据元，通过安全报文传送得以保证。本规范中，MAC 的长度规定为 4 个字节。在安全信息处理过程中用到的 MAC 过程密钥是按照随后描述的过程密钥的产生过程产生的。MAC DEA 密钥的原始密钥用于产生 MAC 过程密钥。

使用单重或三重 DEA 加密方式产生 MAC，需要进行以下几步：

第一步：取 8 个字节的十六进制数字'0'作为初始变量。

第二步：按照顺序将以下数据连接在一起形成数据块：

- CLA，INS，P1，P2，LC①。
- 所有在《中国金融集成电路(IC)卡规范》第 2 部分应用规范中定义的数据。
- 命令的数据域中(如果存在)包含明文或加密的数据。

第三步：将该数据块分成 8 字节为单位的数据块，标号为 D1，D2，D3，D4 等。最后的数据块有可能是 1～8 个字节。

第四步：如果最后的数据块长度是 8 字节的话，则在其后加上十六进制数字'80 00 00 00 00 00 00 00'，转到第五步。

如果最后的数据块长度不足 8 字节，则在其后加上十六进制数字'80'，加上'80'后，如

① LC 表示命令数据域后面 4 个字节 MAC 数据的长度，例如：APPLICATION BLOCK 命令需要产生一个 MAC，计算 MAC 的 LC 的输入值是 4-FE，而不是 0，CLA 包括安全报文的表明('X4')。

果达到 8 字节长度，则转入第五步；否则在其后加入十六进制数字‘0’直到长度达到 8 字节。

第五步：对这些数据块使用 MAC 过程密钥进行加密。如果安全报文传送支持单长度的 MAC DEA 密钥，则依照图 7-11 的方式使用 MAC 过程密钥来产生 MAC。

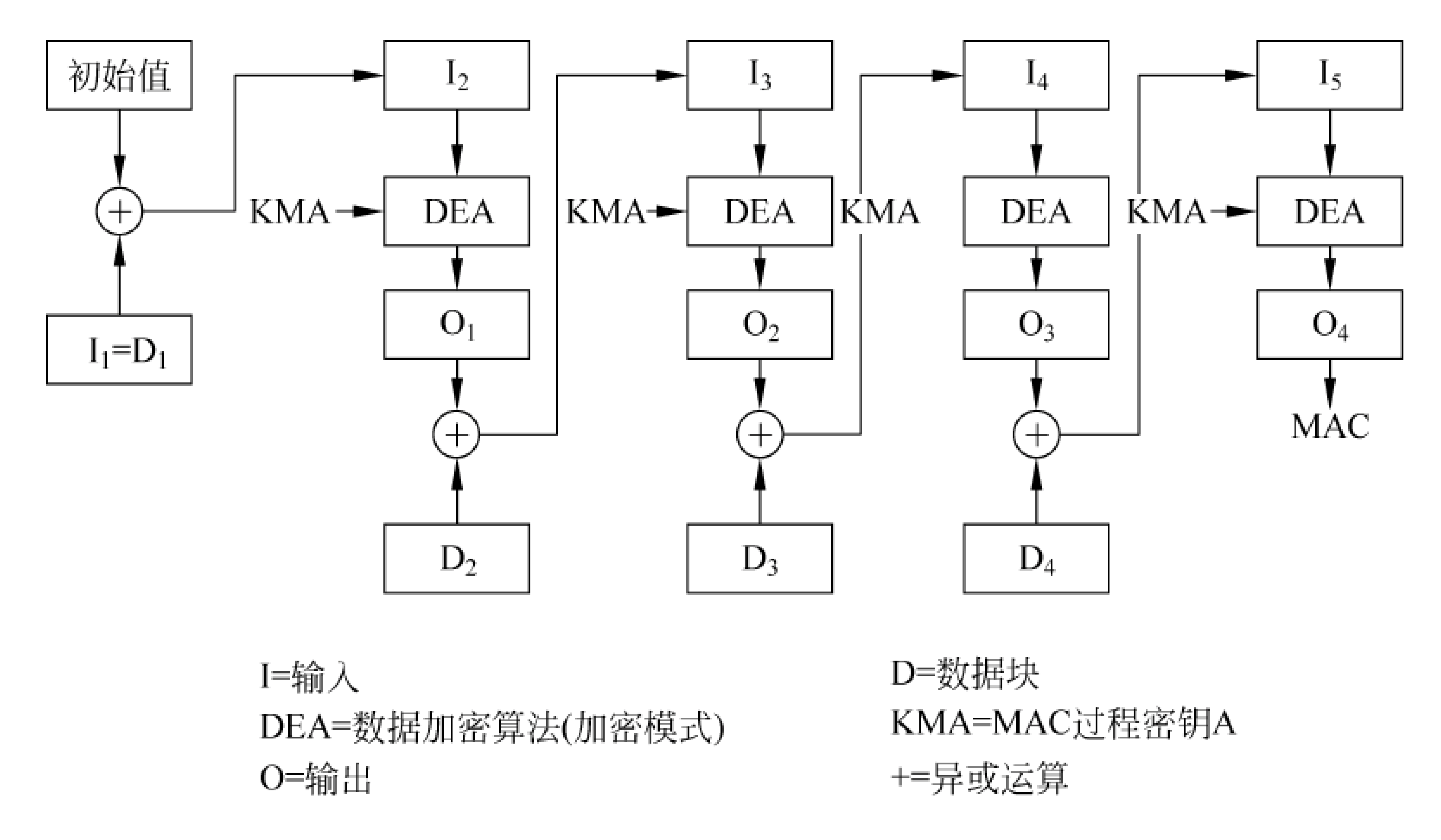


图 7-11　单长度 DEA 密钥的 MAC 算法

如果安全报文传送的处理支持双长度 MAC DEA 密钥，则使用 MAC 过程密钥 A 和 B（MAC 的产生如图 7-12 中所示）。

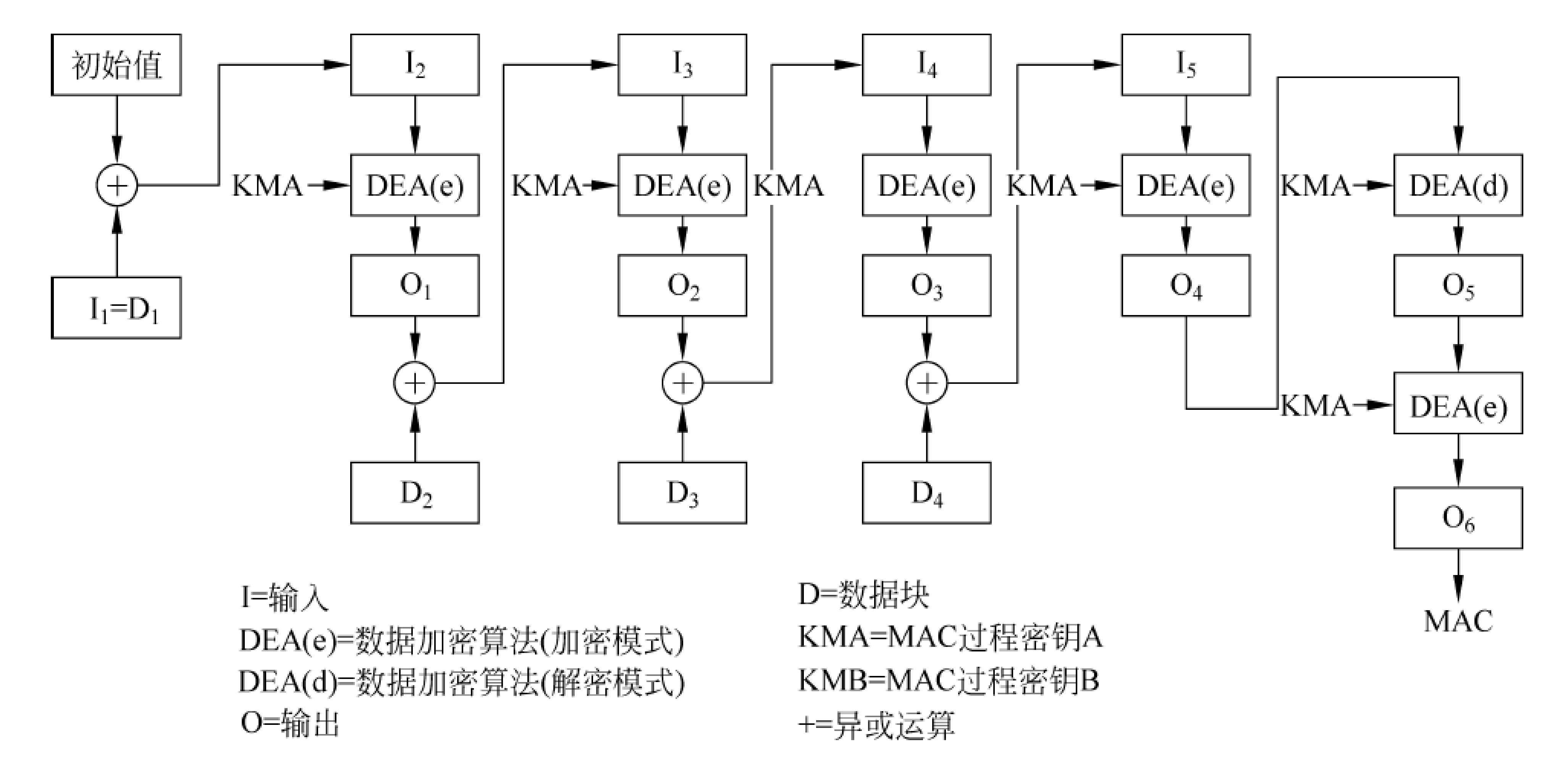


图 7-12　双长度 DEA Key 的 MAC 算法

第六步：最终得到是从计算结果左侧取得的 4 字节长度的 MAC。

3. 数据可靠性

为保证命令中明文数据的保密性，可以将数据加密。所使用的数据加密技术，应被命令发送方和当前卡中被选择的应用所了解。

对安全报文处理过程中用到的数据进行加密，数据加密过程密钥的产生过程是从卡中的数据加密 DEA 密钥开始的。

(1) 当命令中要求的明文数据需要加密时，它先要被格式化为以下形式的数据块：

- 明文数据的长度，不包括填充字符(L_D)。
- 明文数据。
- 填充字符。

(2) 数据加密技术如下所述：

第一步：用 L_D 表示明文数据的长度，在明文数据前加上 L_D 产生新的数据块。

第二步：将第一步中生成的数据块分解成 8 字节数据块，标号为 D_1，D_2，D_3，D_4 等。最后一个数据块长度有可能不足 8 位。

第三步：如果最后(或唯一)的数据块长度等于 8 字节，转入第四步；如果不足 8 字节，在右边添加十六进制数字'80'。添加'80'后，如果长度已达 8 字节，转入第四步；否则，在其右边添加 1 字节十六进制数字'0'直到长度达到 8 字节。

第四步：每一个数据块加密。

如果采用单长度数据加密 DEA 密钥，数据块的加密如图 7-13 所示(使用数据加密过程密钥 A 进行加密)。

如果采用双长度数据加密 DEA 密钥，则数据块的加密如图 7-14 所示(使用数据加密过程密钥 A 和 B 来进行加密)。

第五步：计算结束后，所有加密后的数据块依照原顺序连接在一起(加密后的 D_1，加密后的 D_2 等)，并将结果数据块插入到命令数据域中。

(3) 数据解密计算

卡片接收到命令之后，需要将包含在命令中的加密数据进行解密。数据解密的技术如下。

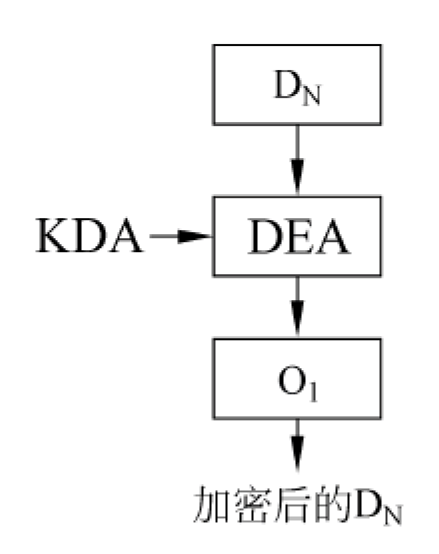


图 7-13 单长度 DEA 密钥的数据加密

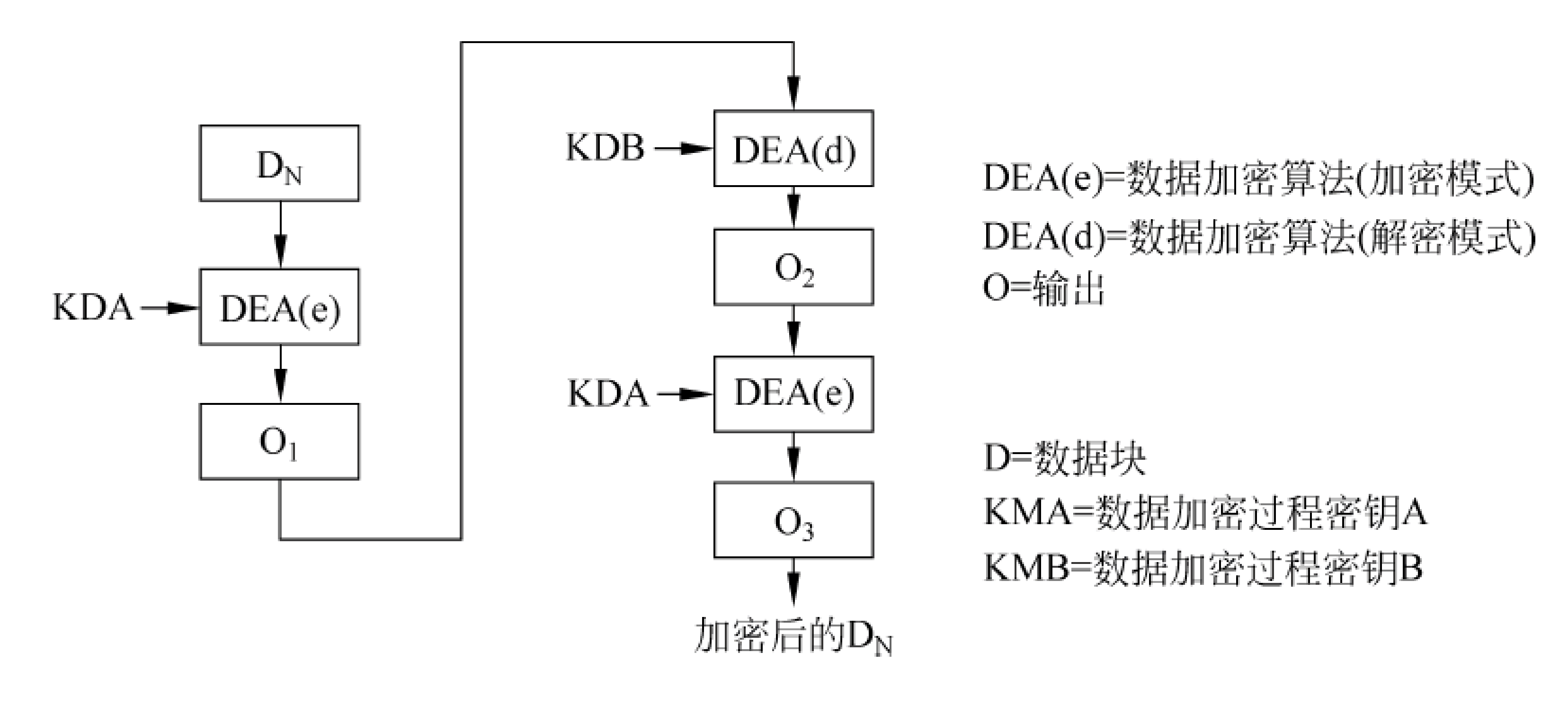


图 7-14 使用双长度 DEA 密钥的数据加密

第一步：将命令数据域中的数据块分解成 8 字节长的数据块，标号为 D_1，D_2，D_3，D_4 等。

如果采用单长度数据加密的 DEA 密钥，数据块解密如图 7-15 所示(使用数据加密过程密钥 A 进行解密)。

如果采用双长度数据加密的 DEA 密钥，则数据块的解密如图 7-16 所示(使用数据加密过程密钥 A 和 B 来进行解密)。

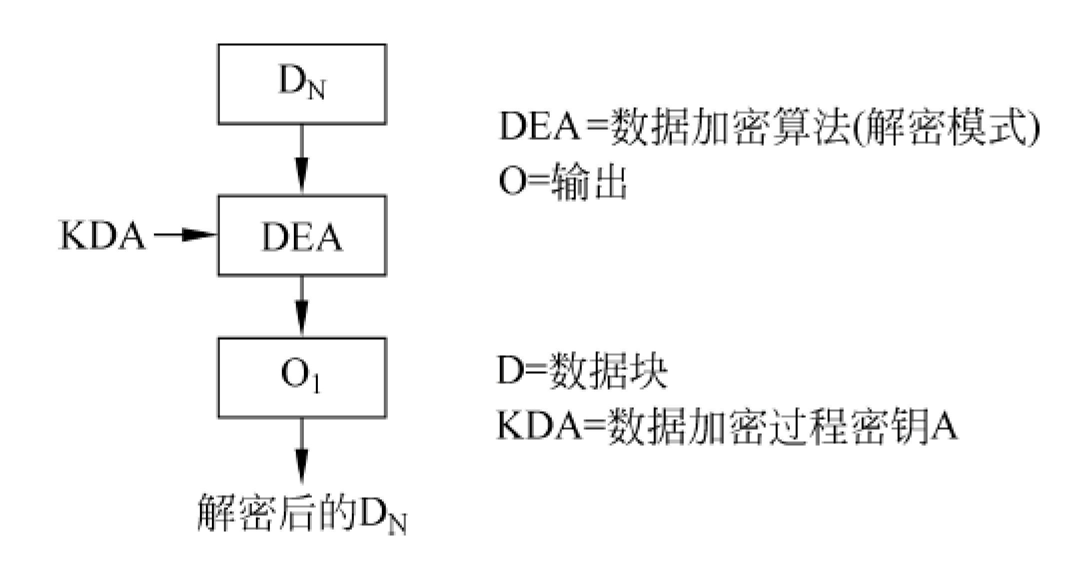


图 7-15 使用单长度 DEA 密钥的数据解密

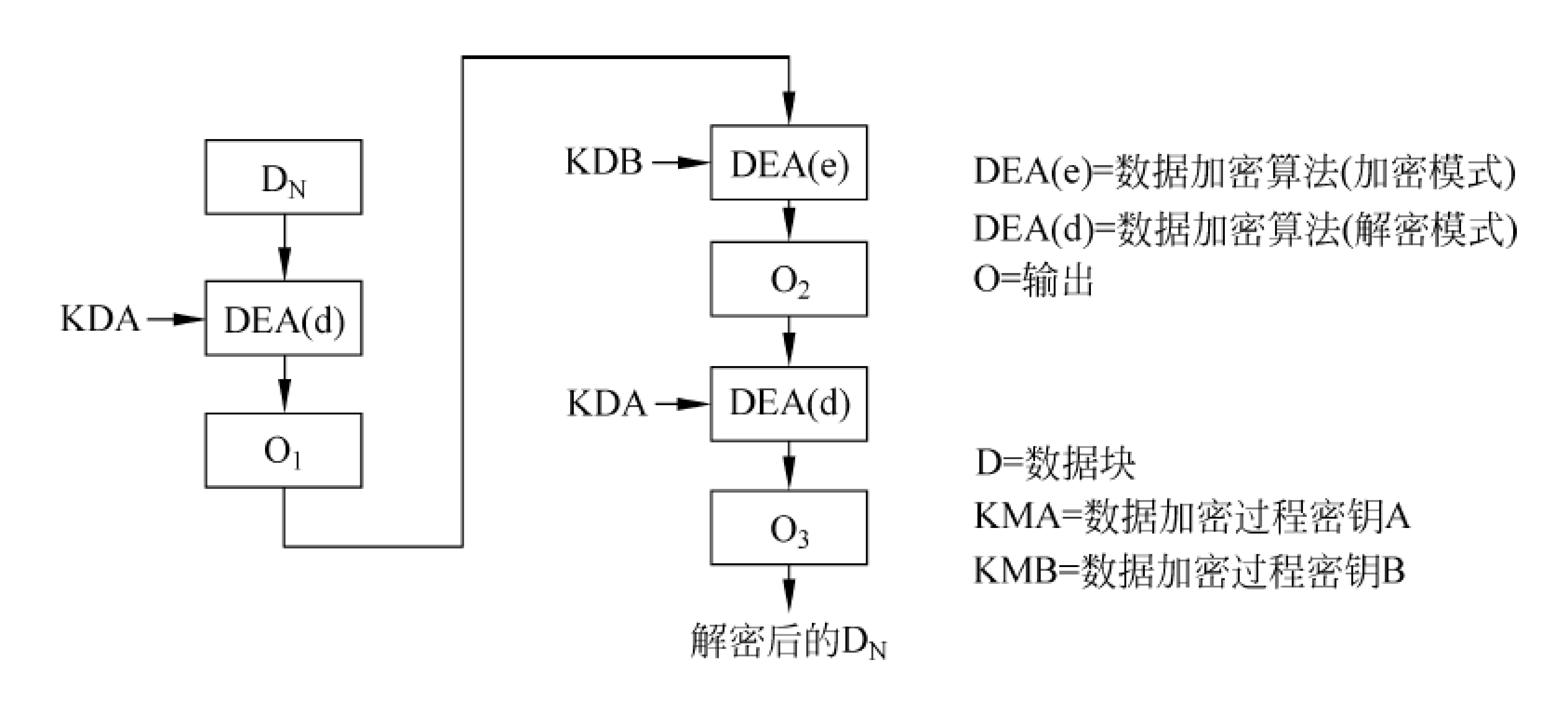


图 7-16 使用双长度 DEA 密钥的数据解密

第二步：计算结束后，所有解密后的数据块依照顺序（解密后的 D_1，解密后的 D_2 等）链接在一起。数据块由 L_D、明文数据、填充字符组成。

第三步：因为 L_D 表示明文数据的长度，因此，它被用来恢复明文数据。

4. 过程密钥的产生

MAC 和数据加密过程密钥的产生如下所述（在本节中统称为“过程密钥 A”和“过程密钥 B”）。

（1）基于单长度 DEA 密钥的过程密钥

第一步：卡片/发卡方决定是使用 MAC DEA 密钥 A 还是数据加密 DEA 密钥 A（后面称为 Key A）来进行所选择的算法处理。

第二步：用 Key A 与预先决定的变量（如当前的交易序号）做异或运算产生过程密钥 A。在做异或运算前，数据（例如：交易序号）如果少于 8 个字节，则在其右边用十六进制数字 0 填满。

（2）基于双长度 DEA 密钥的过程密钥

第一步：卡片/发卡方决定是使用 MAC DEA 密钥 A 和 B（后面称为 Key A 和 Key B）还是数据加密 DEA 密钥 A 和 B 来进行所选择的算法处理。

第二步：用 Key A 与预先决定的变量（如当前的交易序号）做异或运算产生过程密钥 A。在做异或运算前，数据（例如：交易序号）如果少于 8 个字节，则在其右边用十六进制数字 0 填满。

用 Key B 与第二步中产生的过程密钥 A 所用数据的非做异或运算得到过程密钥 B。非运算是以位为单位的，把值为 1 的比特位转换为 0，将值为 0 的比特位转换为 1。在做异

或运算前，数据如果少于 8 个字节，则在其右边用十六进制数字 0 填满。

7.7 标准化组织

为了智能卡行业的健康发展，各个标准化组织制定了相应的一系列标准，为设备制造商提供了一个公平开放竞争的市场环境，同时也确保了技术的国家和国际间的互用性和兼容性，接下来简单介绍几个标准化组织。

(1) ISO

国际标准化组织(International Organization for Standardization，ISO)是目前世界上最大、最有权威性的国际标准化专门机构。1946 年 10 月 14 日至 26 日，中、英、美、法、苏等 25 个国家的 64 名代表集会于伦敦，正式表决通过建立国际标准化组织。1947 年 2 月 23 日，ISO 章程得到 15 个国家标准化机构的认可，国际标准化组织宣告正式成立。参加 1946 年 10 月 14 日伦敦会议的 25 个国家为 ISO 的创始人。ISO 是联合国经社理事会的甲级咨询组织和贸发理事会综合级(即最高级)咨询组织。此外，ISO 还与 600 多个国际组织保持着协作关系。

国际标准化组织的目的和宗旨是："在全世界范围内促进标准化工作的发展，以便于国际物资交流和服务，并扩大在知识、科学、技术和经济方面的合作。"其主要活动是制定国际标准，协调世界范围的标准化工作，组织各成员国和技术委员会进行情报交流，以及与其他国际组织进行合作，共同研究有关标准化问题。

(2) IEC

国际电工委员会(the International Electro Technical Commission，IEC)成立于 1906 年，至今已有 90 多年的历史。它是世界上成立最早的国际性电工标准化机构，负责有关电气工程和电子工程领域中的国际标准化工作。

IEC 的宗旨是："促进电气、电子工程领域中标准化及有关问题的国际合作，增进国际间的相互了解。"为实现这一目的，IEC 出版包括国际标准在内的各种出版物，并希望各成员在本国条件允许的情况下，在本国的标准化工作中使用这些标准。近 20 年来，IEC 的工作领域和组织规模均有了相当大的发展。今天 IEC 成员国已从 1960 年的 35 个增加到 61 个。它们拥有世界人口的 80%，消耗的电能占全球消耗量的 95%。目前 IEC 的工作领域已由单纯研究电气设备、电机的名词术语和功率等问题扩展到电子、电力、微电子及其应用、通信、视听、机器人、信息技术、新型医疗器械和核仪表等电工技术的各个方面。IEC 标准已涉及了世界市场中的 35%的产品，到 20 世纪末，这个数字可达 50%。

IEC 标准的权威性是世界公认的。IEC 每年要在世界各地召开 100 多次国际标准会议，世界各国的近 10 万名专家在参与 IEC 的标准制、修订工作。IEC 现在有技术委员会 89 个、分技术委员会 88 个。IEC 标准在迅速增加，1963 年只有 120 个标准，截止到 2001 年 12 月底，IEC 已制定了 5098 个国际标准。

(3) ITU

国际电信联盟(International Telecommunication Union，ITU)于 1865 年 5 月在巴黎成立，1947 年成为联合国的专门机构。

ITU 是世界各国政府的电信主管部门之间协调电信事务的一个国际组织，它研究制定

有关电信业务的规章制度,通过决议提出推荐标准,收集有关情报。ITU 的目的和任务是:"维持和发展国际合作,以改进和合理利用电信,促进技术设施的发展及其有效运用,以提高电信业务的效率,扩大技术设施的用途,并尽可能使之得到广泛应用,协调各目的活动。"

(4) NIST

美国国家标准与技术研究院(National Institute of Standards and Technology,NIST)隶属于美国商务部,下有八个实验室和一个研究中心,主要从事物理、生物和工程方面的基础和应用研究,以及测量技术和测试方法方面的研究,并提供标准、标准参考数据及有关服务,在国际上享有很高的声誉。NIST 作为美国国家计量院,负责建立和维护用户复现国际单位制基本单位和 SI 单位制的美国国家计量标准,以联邦信息处理标准(Federal Information Processing Standards,FIPS)的形式发布标准。从 1977 年公布的数据加密标准 DES 开始,NIST 就制定了一系列有关密码技术的 FIPS,在技术规范的前提下对密码产品进行严格的检验。

(5) ANSI

美国国家标准学会(American National Standards Institute,ANSI)是非盈利性质的民间标准化团体,但它实际上已成为美国国家标准化中心,美国各界标准化活动都围绕其进行。通过它,可使政府有关系统和民间系统相互配合,以发挥政府和民间标准化系统之间的桥梁作用。ANSI 协调并指导美国全国的标准化活动,给标准制定、研究和使用单位以帮助,提供国内外标准化情报,同时又起着行政管理机关的作用。

(6) IEEE

美国电气电子工程师学会(Institute of Electrical and Electronics Engineers,IEEE)于 1963 年由美国电气工程师学会(AIEE)和美国无线电工程师学会(IRE)合并而成,是美国规模最大的专业学会。它由大约 17 万名从事电气工程、电子和有关领域的专业人员组成,分设 10 个地区和 206 个地方分会,并设有 31 个技术委员会。IEEE 制定的标准内容有电气与电子设备、试验方法、元器件、符号、定义以及测试方法等。

(7) EMV

EMV 组织由世界主要信用卡联合体 Visa、MasterCard 和 Europay 于 1993 年创立,旨在共同制定的金融支付系统的 IC 卡及读卡终端的互操作规范。EMV 标准是 EMV 组织制定并着眼于取代磁条卡,实现全球范围金融 IC 卡的跨国界、跨厂商、跨金融机构的互操作,并提供一卡多用的基础。EMV 规范集中了已有的相关金融规范的优点,安全机制更完备。

目前,各国基本上都遵循 EMV 规范,正在积极推行由磁条卡向 IC 卡的迁移(简称 EMV 迁移)。

(8) SAC

中国国家标准化管理委员会(Standardization Administration of the People's Republic of China,SAC)为国家质检总局管理的事业单位。国家标准化管理委员会是国务院授权的履行行政管理职能,统一管理全国标准化工作的主管机构。

中国国家标准化管理委员会的主要职责是:参与起草、修订国家标准化法律、法规的工作;拟定和贯彻执行国家标准化工作的方针、政策;拟定全国标准化管理规章,制定相关制度;组织实施标准化法律、法规和规章、制度。负责制定国家标准化事业发展规划;负责组织、协调和编制国家标准(含国家标准样品)的制定、修订计划。负责组织国家标准的制定、

修订工作，负责国家标准的统一审查、批准、编号和发布等工作。

参考文献

[1] 戴节永，周方，黄逸之. PKCS＃11 密码令牌接口标准技术综述. 网络安全技术与应用，2008 年，(11)：35～37
[2] 中国金融集成电路(IC)卡电子钱包/电子存折规范 V2.0 第一部分：卡片规范，中国金融集成电路(IC)卡标准修订工作组，2004
[3] International Standard ISO/IEC 15946，Information technology. Security techniques，Cryptographic techniques based on elliptic curves，2002
[4] International Standard ISO/IEC 9796-2，Information technology. Security techniques，Digital signature schemes giving message recovery，Part 2：Integer factorization based mechanisms，2002
[5] International Standard ISO/IEC 7816-8，Identification cards. Integrated circuit cards，Part 8：Commands for security operation，2004
[6] 刘凤. 国外电子商务标准发展概况. 中国质量技术监督，2005，(10)：56～57
[7] 周建中. 美国标准与技术研究院绩效评估的实践、方法及启示. 中国科技论坛，2009，(01)：135～139
[8] 宣湘. 走在世界计量前沿的美国国家标准技术研究院. 中国计量，2005，(06)：46～48
[9] 朱建新，朱立国. 基于 EMV 标准的金融 IC 卡安全框架设计与实现，微计算机信息，2007，23(10.3)：65～67
[10] GlobalPlatform. Card Specification Version2.2，2006
[11] GlobalPlatform Card Technology. Secure Channel Protocol 03，Card Specification V2.2-Amendment D，Version 0.80，2009
[12] 刘慧. Java 智能卡的安全性分析与研究. 山东：山东大学硕士学位论文. 2008
[13] 臧宏伟. JavaCard Applet 开发讲义. 北京握奇数据系统有限公司



低层设计

智能卡系统组成主要包括五层：应用层、服务层、驱动层、硬件层、嵌入式软件，其结构如图 8-1 所示。智能卡系统良好的设计结构对代码的复用和移植均起到重要意义。智能卡由专业芯片设计公司设计，并由专业封装公司进行封装，属于微电子领域的范畴，非著者所熟悉的专业，所以本书不进行描述，可参考相关书籍。智能卡芯片操作系统是嵌入在芯片中的嵌入式监控软件。机具是连接智能卡和应用程序的桥梁，机具驱动则把应用层和底层模块需要发生变化的部分同可以复用的代码部分分开，并支持 PC/SC 机具和非 PC/SC 机具的代码分开。智能卡加密服务提供者作为服务层，内容较多，所以单列一章，放在本书第 9 章。应用系统是符合某业务要求的人机界面和业务功能的实现，则在第 10 章进行介绍。

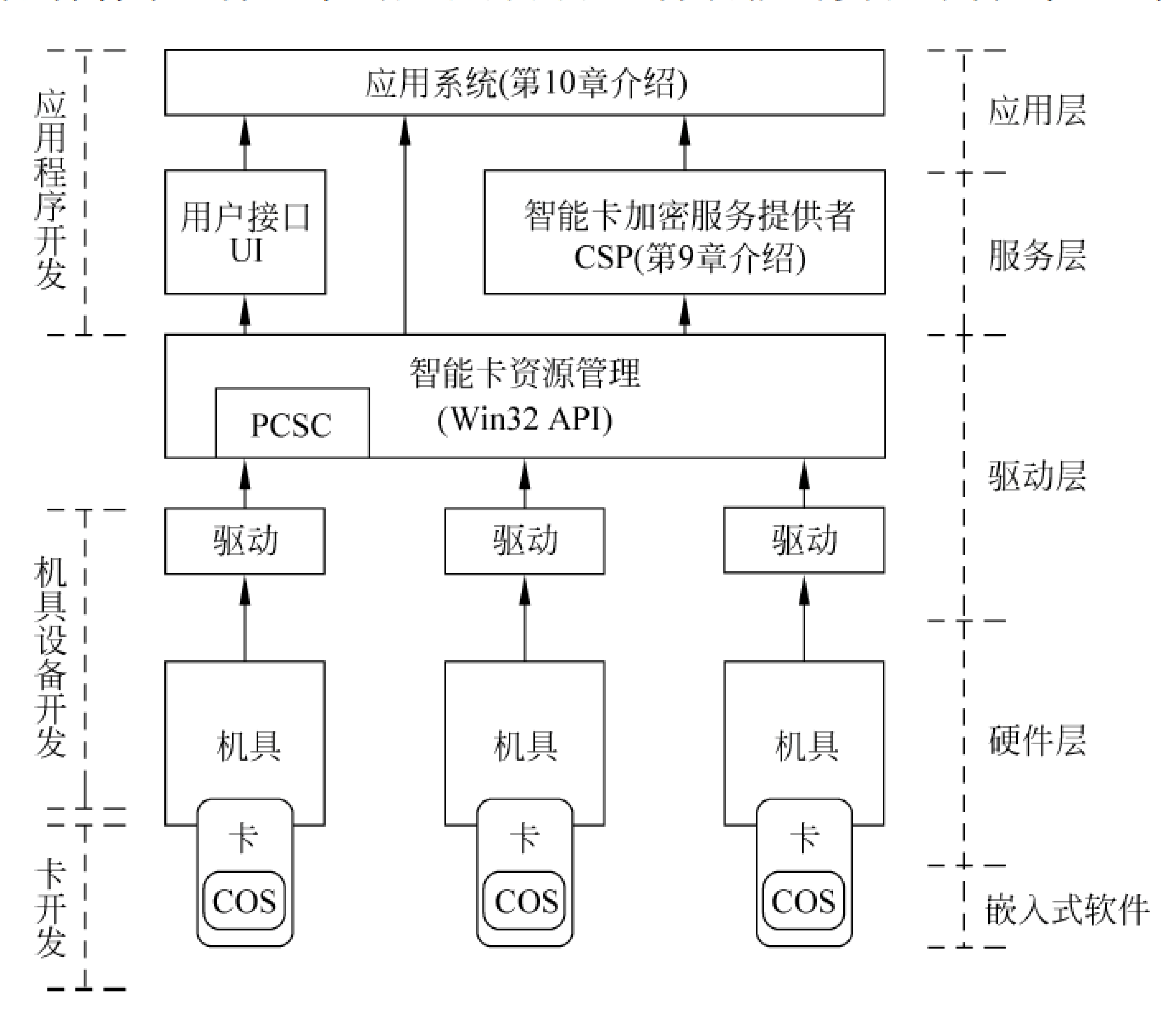


图 8-1　智能卡应用系统架构图

本章按照从底向上的顺序，主要描述了芯片操作系统的设计、接触式/非接触式读写设备的设计和应用程序接口的设计。

8.1　芯片操作系统设计

芯片操作系统(chip operating system，COS)是嵌入在智能卡芯片内，管理硬件资源、执行数据的安全存取并与外部接口设备通信的监控软件系统。它是智能卡资源的管理者和安

全保密的基础，其主要功能是控制智能卡与外界的信息交换，管理智能卡内的存储器并在卡内部完成各种命令的处理。

8.1.1 设计原则

COS软件设计首先要有一个良好的结构。分层结构则为一种良好的设计结构。在这种设计结构中，COS分成主要四层：应用层、功能引擎层、硬件接口层、硬件资源模块层。其中功能引擎层又可分为安全管理、文件管理、命令管理和通信管理四个模块。把应用层和底层模块需要发生变化的部分同可以复用的代码部分分开，多款芯片上应用COS，只需移植与硬件相关的底层模块层代码即可。COS软件层次图如图8-2所示。

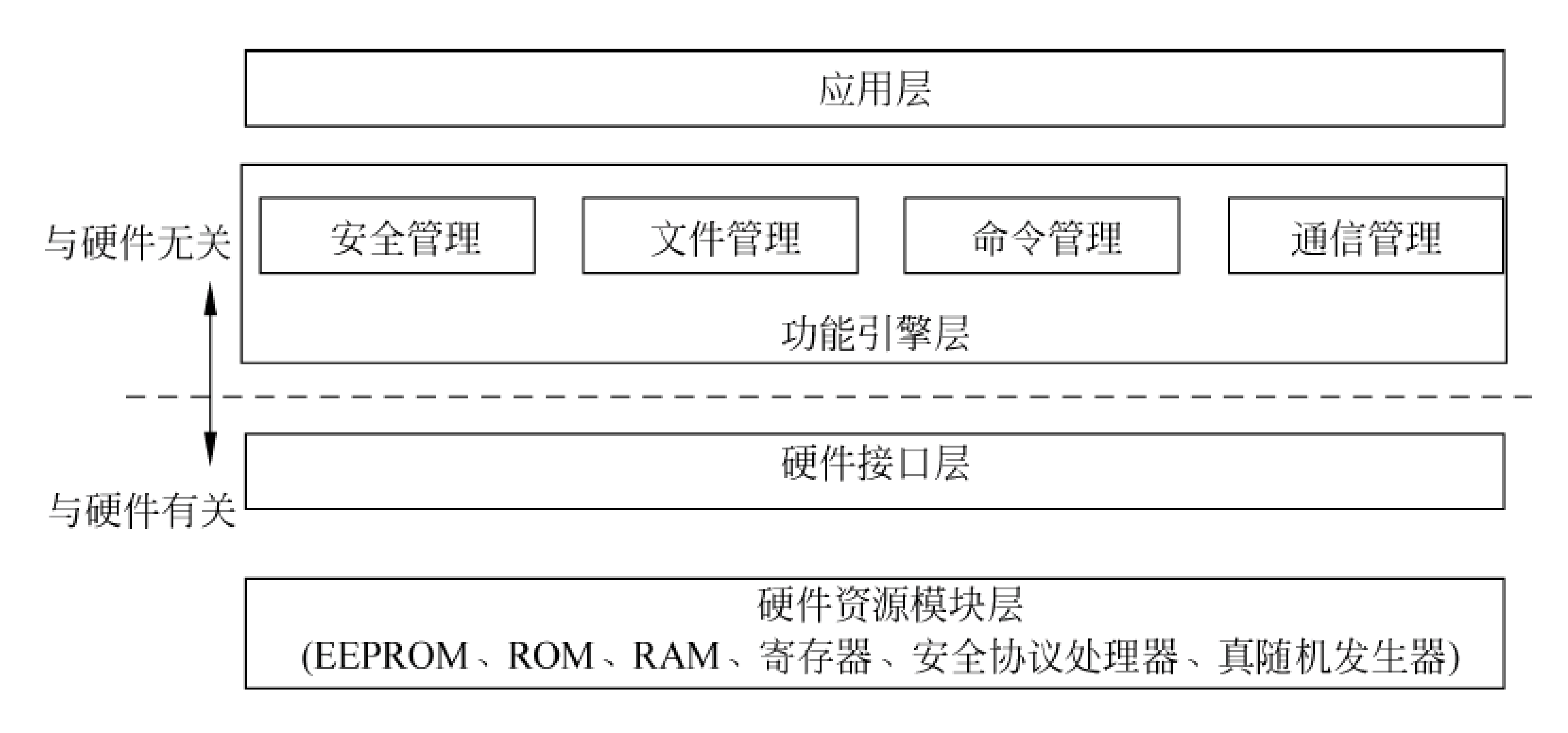


图8-2 COS软件层次图

在分层设计的同时，将功能引擎层实施模块化处理。COS软件的各模块组成和数据处理流程如图8-3所示。外部信息通过通信管理模块的传送进入COS之后，首先将接受安全管理模块对它进行信息的合法性检查；其后要接受对输入信息内容的可执行性判断，这由命令管理模块执行；最后COS根据合法且有效的信息对芯片中的EEPROM执行相应的操作(EEPROM中的数据以文件形式存放的)，该操作由文件管理模块实现。

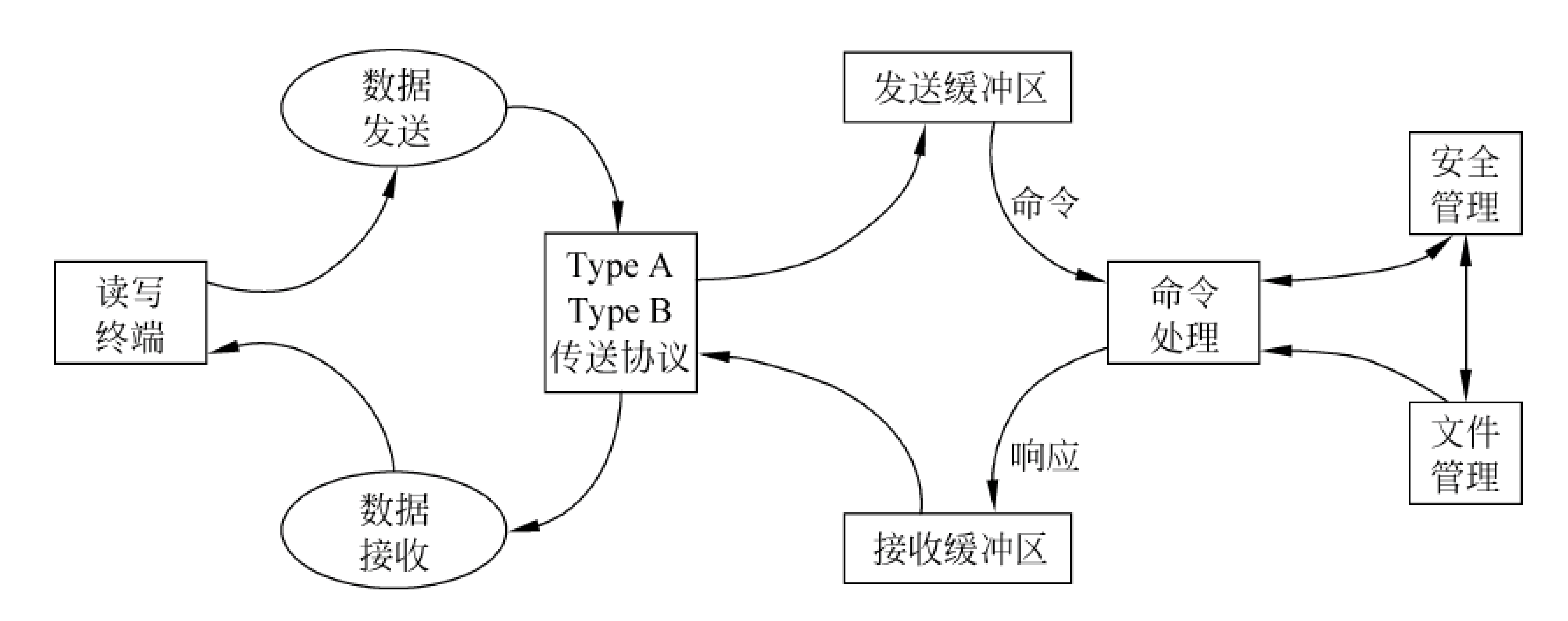


图8-3 COS软件模块图

在COS设计和实施的过程中，必须充分考虑可移植性的问题，尽量减少以后COS升级和向其他硬件平台移植的工作量。

在设计过程中，必须遵循以下原则：

(1) 层次划分清楚；

(2) 与硬件相关的部分与其他部分相对独立;

(3) 尽可能减少不同功能之间的关联;

(4) 尽量减少模块之间的关联;

(5) 模块的功能在满足现有的需求下,应尽可能扩充。

8.1.2 功能模块

COS 系统功能实现,可分为 4 个主功能模块:通信管理模块、命令管理模块、安全管理模块和文件管理模块。

(1) 通信管理模块

通信管理模块主要是依据智能卡所使用的信息传输协议,对读写设备发出的命令进行接收。同时,把对命令的响应按照传输协议的格式发送出去。这一部分主要和智能卡具体使用的通信协议有关,而且,所使用的通信协议越复杂,这一部分的实现就越复杂、越困难。电子护照中通信模块中使用的数据传输协议为非接触式协议,符合 ISO/IEC 14443 Type A/B 标准。

(2) 命令管理模块

COS 通过通信管理模块接收命令,然后由命令管理模块对接收到的每条命令的命令头做语法分析,分析和检查命令参数的正确性,如果参数不正确将返回过程字节给通信管理模块,然后根据每条命令分别执行安全管理模块和文件管理模块的相应子模块。命令执行完后,负责将安全管理模块的应答数据返回给通信管理模块,并最终由通信管理模块发送给读写设备。

(3) 文件系统模块

文件系统是遵照 ISO/IEC 7816-4 协议来组织的,具体的层次结构如图 8-4 所示。

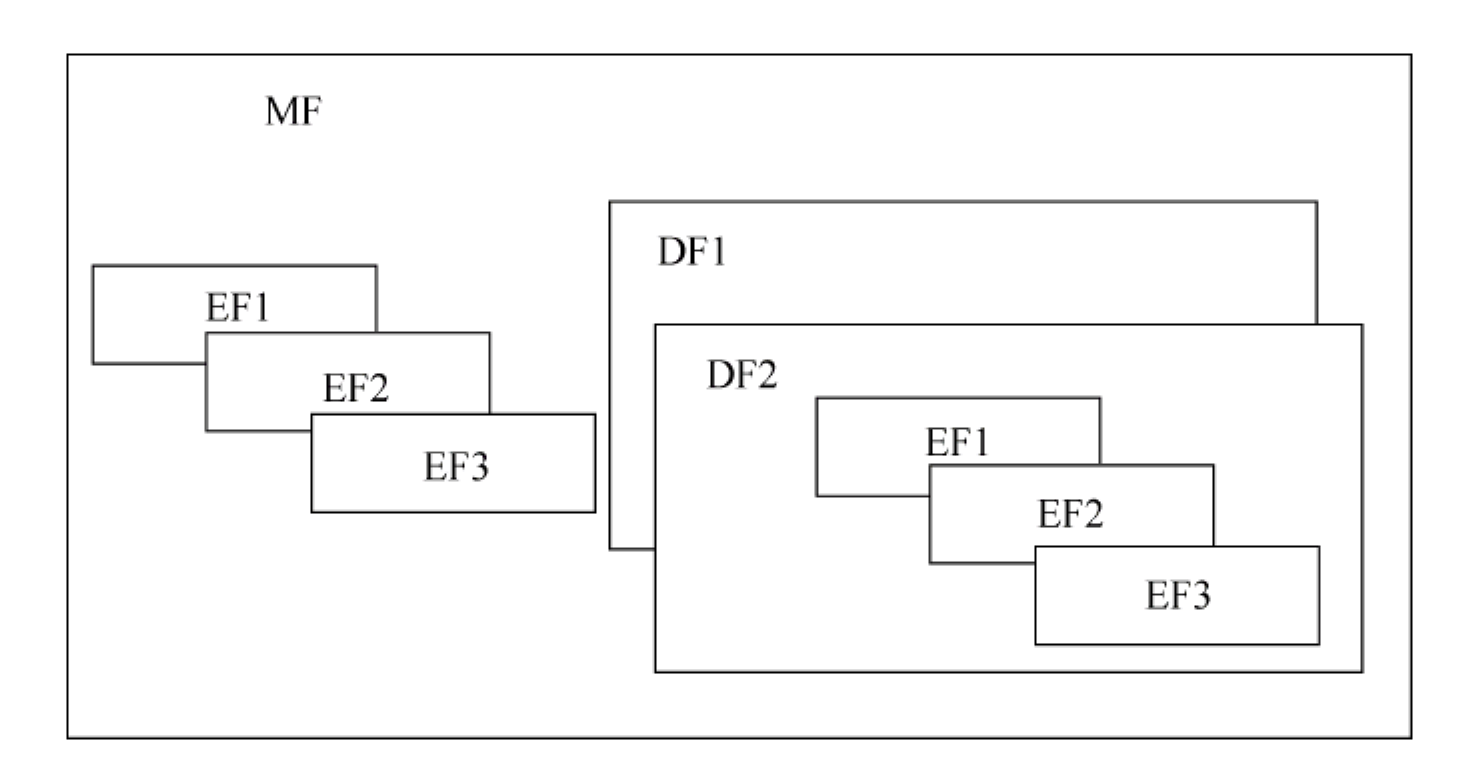


图 8-4 文件系统层次图

主控文件(master file,MF)是整个文件系统的根(可看作根目录),每个电子护照有且仅有一个主控文件。主控文件占有的存储空间包括 MF 文件头的大小以及 MF 所管理的 EF 和 DF 的存储空间。

专用文件(dedicated file,DF)是在 MF 下针对不同的应用建立起来的一种文件,是位于 MF 之下的含有 EF 的一种文件结构,它存储了某个应用的全部数据以及与应用操作相关的安全数据。

基本文件(elementary file,EF)存储了各种应用的数据和管理信息,它存在于 MF 和 DF 之下。

(4) 安全管理模块

智能卡之安全体系主要包括三大部分:安全条件、安全状态和安全机制。

安全条件主要包括命令执行条件、卡片生命周期和文件操作权限等。命令执行条件指命令执行时需要的前提条件,比如外部认证前需执行获取新随机数命令。

安全状态是智能卡在当前所处的一种状态,智能卡处理完某命令之后得到的安全状态常可以利用智能卡在当前已经满足的安全条件的集合来表示。

安全机制可以认为是安全状态实现转移所采用的转移方法和手段。在本书的第 5 章进行了详细描述。

8.1.3 文件结构

对于应用资源非常紧张的 EEPROM 存放数据而言,无疑它的存储分配是最重要的。应用数据在 COS 中以文件的形式存储在芯片的 EEPROM 中。智能卡 COS 的文件系统采用与 ISO/IEC 7816-4 标准相同的树形结构。

文件的物理组织涉及一个文件在存储设备上是如何放置的。它和文件的存取方法有密切关系,另外也取决于存储设备的物理特性。从逻辑上讲,所有的文件必须是连续的,这是为了方便寻址,这就要求文件要以链的形式存在。

文件的创建关系到 COS 对文件系统的安排。通过文件系统,COS 将存放在 EEPROM 里的数据有条理地组织在一起,从而可以方便地对数据进行管理。文件系统的结构由 COS 创建文件的方法来决定。目前 COS 中常用的文件分配方法包括顺序式文件结构、链表式文件结构等多种方法。

1. 顺序式文件结构

把主文件(MF)看作根目录,它下面还有第二级目录,基本文件存在于 MF 和 DF 下。MF 和 DF 都有自己的 FAT 表,FAT 表预设 16 个节点,每一个节点与之下的一个文件相对应。具体文件结构如表 8-1 所示。

表 8-1 顺序式文件结构表

MF 文件部分	MF 文件头(10H 字节)
	MF 文件下的 FAT 表,共分为 16 项,每一项对应之下的一个 EF 或 DF
	MF 下的 Key 文件(10H 字节的文件头和相应的文件体)
	MF 下的其他 EF 文件(10H 字节的文件头和响应的文件体)
MF 文件下的 DF1 文件	MF 下的 DF 文件 DF1 的文件头(10H 字节)
	DF1 文件下的 FAT 表,共 16 项,对应于之下的最多 16 个 EF 文件
	DF1 下的 Key 文件(10H 字节的文件头和相应的文件体)
	DF1 下的其他 EF 文件(10H 字节的文件头和相应的文件体)
MF 文件下的 DF2 文件	MF 下的 DF 文件 DF2 的文件头(10H 字节)
	DF2 文件下的 FAT 表,共 16 项,对应于之下的最多 16 个 EF 文件
	DF2 下的 Key 文件(10H 字节的文件头和相应的文件体)
	DF2 下的其他 EF 文件(10H 字节的文件头和相应的文件体)
MF 文件下的其他 DF 文件	…

由于没有采用链表的形式,所以一旦删除 DF 文件或者 EF 文件,它所占用的空间将没法被重新利用,并且受到 FAT 表固定大小的限制,MF 和 DF 文件下最多只能安排 16 个文件。当时设计出这样一个文件结构主要是出于对空间利用的考虑,但是却使得该系统的灵活性不够,显得比较死板。针对这些缺点,可以有一个改进方案,见下一种文件结构。

2. 链表式文件结构

为了提高文件系统的灵活性,我们引入了文件链表的概念,在 IC 卡中,每次最多可以对 64 字节的 EEPROM 进行写操作,因此将数据块大小定义为 64 字节,每个数据块的最后两个字节作为链接下一个数据块的指针,这样将有 2/64=1/32 的 EEPROM 空间被用作链接指针,也就是说 16KB 的 EEPROM 将有 0.5KB 被浪费,毫无疑问,这样的开销太大了。所以不能采用这种数据块的形式,也就是不引入 PC 下操作系统中"簇"的概念。

在 IC 卡内,文件的平均大小都很小,对文件的删除增加的操作不是很频繁,并且 EEPROM 的空间异常珍贵,这些都决定了 IC 卡的文件系统不能照搬大型操作系统的文件系统,必须有所改动。

因此,考虑到文件系统的灵活性,要采用文件链表,针对 IC 卡的特殊性,不采用固定大小的数据块。

(1) MF 下的文件结构

在实际存储时,MF 下的每个文件可以连续存储在一起,也可以分成几个部分分开存储,即文件链表。分开存储时,各个部分的第一个字节表示本存储部分的大小,最后两个字节存放指针,指向下一个部分。MF 下的整个存储空间全部以文件的形式进行管理,没有被占用的空闲存储空间也被看成一个文件,在 MF 下增加文件时,从这里申请空间,删除文件时,空间被释放到这里。

MF 下可以包含 DF 和 EF 文件,每一个 DF 或者 EF 都对应 MF 的 FAT 表中的一项。FAT 表记录该文件的文件标识符、起始地址和文件类型(EF 还是 DF)。MF 下的 FAT 表大小不固定,可以根据需要自动增大或减小。但为了便于操作,一次增加或减少的大小固定,为 8 个 FAT 表数据项。

(2) DF 下的文件结构

DF 下的文件结构与 MF 下的文件结构几乎完全一样,只是在 DF 内要记录父目录的地址信息。DF 对其下的文件独立管理,MF 不进行任何干预,每一个 DF 下都可以有 EF 和 DF,文件个数可以按照需要,以 8 为单位增加或减少。

(3) EF 文件的存储形式

为了实现的方便,EF 文件不能采用链接的形式进行存储。同样 MF 以及 DF 的文件头和 FAT 表单元本身都不可以分开存储。

按照这种方案实现文件系统,理论上讲,目录可做到无限级,目录下的文件个数可达无限个,它们只受 EEPROM 空间的限制。

8.1.4 EF 分类

根据 ISO/IEC 7816-4 有关基本文件结构的定义,这里介绍 4 种基本文件(EF)的结构。

(1) 二进制文件

二进制文件在 EEPROM 中为用户开辟一个连续空间,用户可以在这个空间自由的安

排自己的数据。二进制文件数据以字节为单位进行读写，数据结构根据应用进行定义。二进制文件结构见图 8-5 所示。

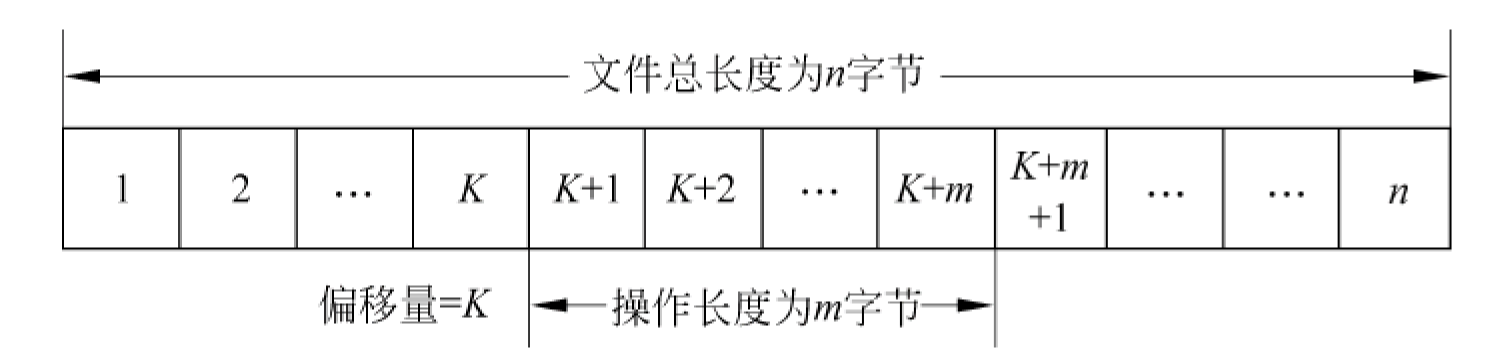


图 8-5　二进制文件结构

（2）线性定长记录文件

定长记录文件在 EEPROM 中为每一个记录开辟一个定长的连续空间，用户可以在这些空间内自由安排自己的数据。但是读、写、添加时必须按记录为单元进行操作。定长记录文件的使用空间在创建文件时分配，在使用过程中不能改变其大小。线性定长记录文件结构见图 8-6 所示。

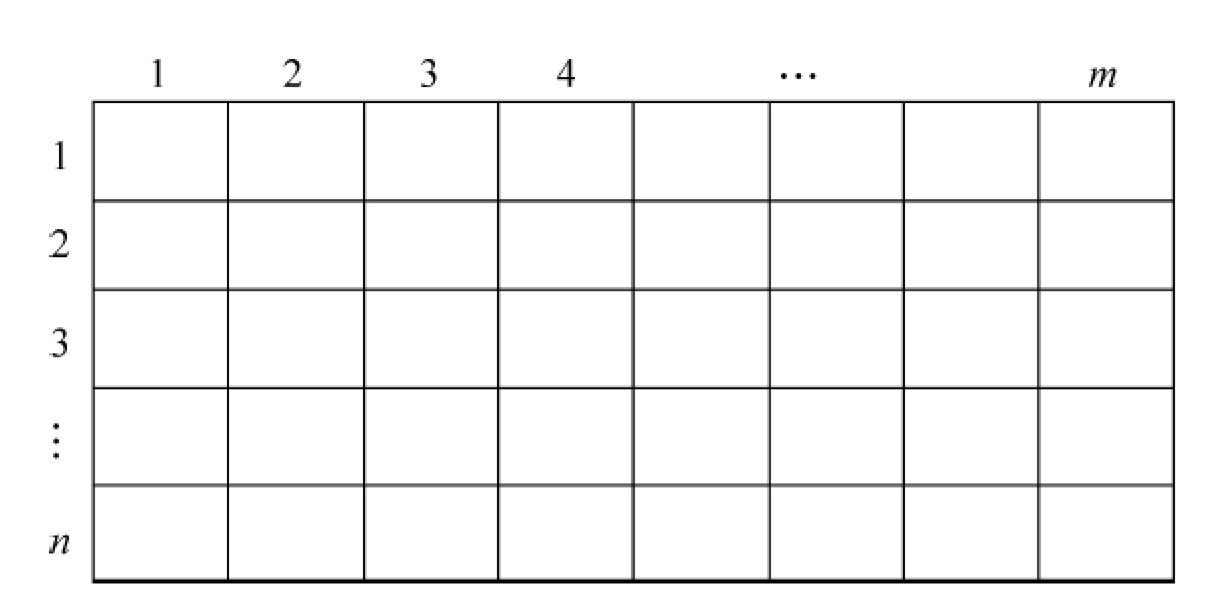


图 8-6　线性定长记录文件结构

（3）线性变长记录文件

在这类结构中，每条记录的长度可以各不相同，仍然是通过记录号来访问各条记录。在读取和修改记录时，操作与线性定长记录的操作相同。线性变长记录文件结构见图 8-7 所示。

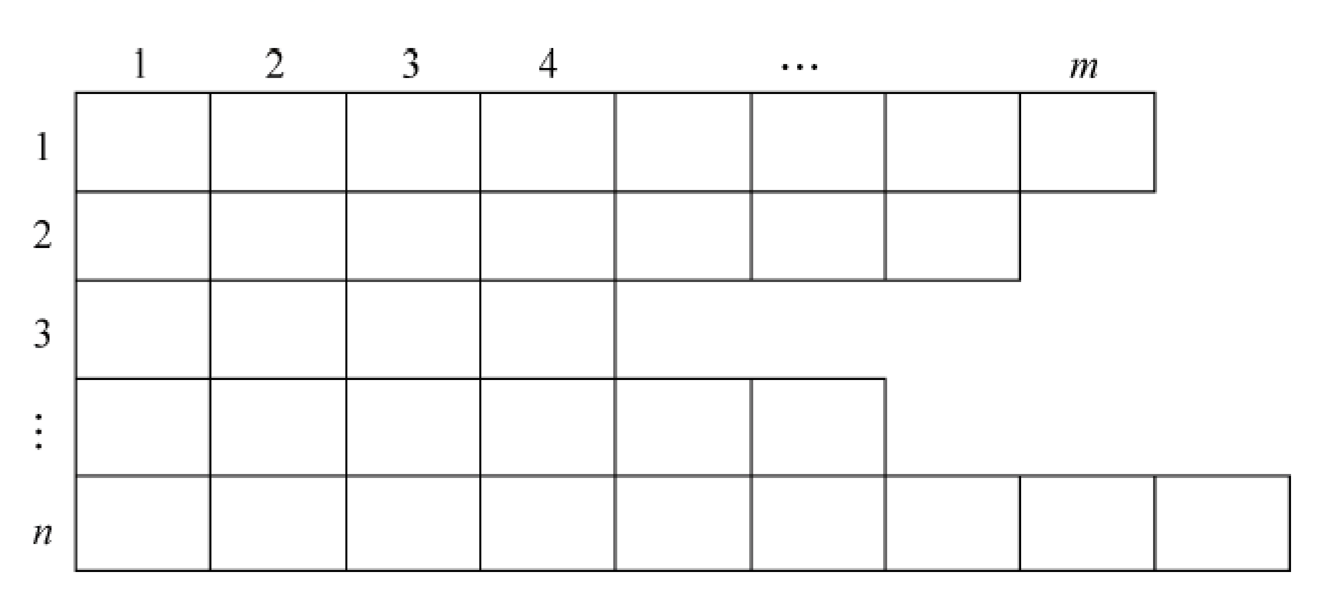


图 8-7　线性变长记录文件结构

（4）循环定长记录文件

这是一类特殊的定长记录文件结构。在逻辑上，这类文件可看作一个环形记录队列，记录按照先进先出的原则存储。添加记录时，最新一次写入记录的记录号为 1，上一次写入记录的记录号为 2，依次类推。记录的个数与预留的记录空间大小以及记录的长度相关，记录个数＝记录空间大小整除记录长度。循环定长记录文件结构见图 8-8 所示。

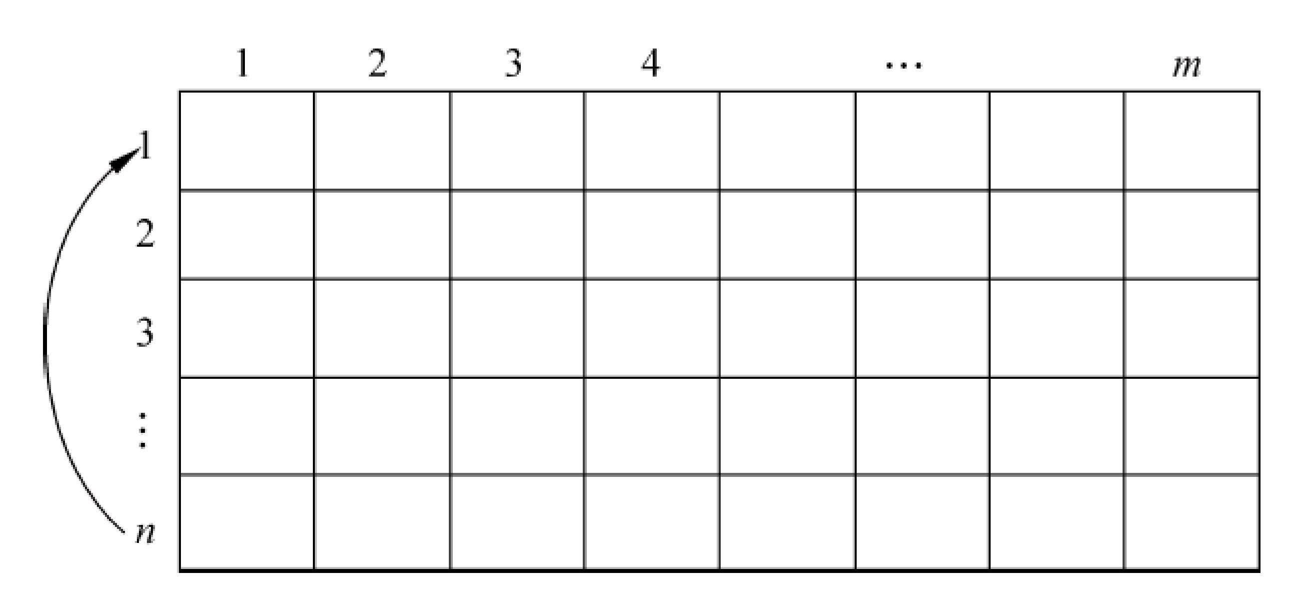


图 8-8 线性循环记录文件结构

此外还有一些只能特殊使用的文件类型，如钱包文件、存折文件、密钥文件等，但其文件结构也不超出以上 4 种文件的类型。文件管理模块完成所有与文件操作有关的工作，如文件的创建、删除、读、写等，同时文件管理模块还负责对这些操作权限的检验。

8.1.5 命令格式

1. 命令响应对

在读写终端和智能卡之间的信息交换是命令-应答对结构。读写终端发送一个命令到智能卡，智能卡执行命令并将响应信息返回给读写终端。

按 ISO/IEC 7816 有关标准定义，一个应用协议数据单元(APDU)或者含有命令信息或者含有响应信息，APDU 可以理解为读写终端与智能卡之间一次通信传输的最小信息单位。

按 ISO/IEC 7816 有关标准，信息结构有两种：一是命令信息结构，二是响应信息结构。命令信息结构符合 ISO/IEC 7816-4 标准，包括命令 APDU 和响应 APDU。其中命令 APDU 包含两部分：固定的四个字节命令头和长度可变的命令体，如图 8-9 所示。

响应 APDU 由数据字段和状态字段两部分组成，参见图 8-10。

命令头	命令体
CLA INS P1 P2	[LC] [数据域] [LE]

图 8-9 命令 APDU 结构图

数据字段	状态字段
LE 字节的数据域	SW1 SW2

图 8-10 响应 APDU 的结构

命令和响应是成对出现的，即从读写器向卡发送一个命令 APDU，会从卡向读写器发回一个相应的响应 APDU。表 8-2 列出了命令 APDU 和响应 APDU 中各字段表示的含义和字节数。

表 8-2 命令响应对

字 段	描 述	字 节 数
命令 APDU		
命令头	类别字节 CLA	1
	指令字节 INS	1
	参数字节 P1-P2	2
LC 字段	命令报文数据域长度	0,1 或 3
命令数据字段	命令数据	LC
LE 字段	期望卡返回的最大数据长度	0 或 1

续表

字　段	描　述	字　节　数
响应 APDU		
数据字段	响应数据	LE
响应尾标	状态字节 SW1-SW2	2

在具体应用中，根据命令 APDU 和响应 APDU 的报文结构组合，共有 4 种情况，如图 8-11 所示。

情形1：

命令：	CLA	INS	P1	P2	
响应：	SW1	SW2			

情形2：

命令：	CLA	INS	P1	P2	LE	
响应：	LE字节的DATA				SW1	SW2

情形3：

命令：	CLA	INS	P1	P2	DATA
响应：	SW1	SW2			

情形4：

命令：	CLA	INS	P1	P2	LC	DATA	LE
响应：	LE字节的DATA				SW1	SW2	

图 8-11　命令 APDU 和响应 APDU 的报文结构组合

2. 指令类别字节 CLA

CLA 字节指示命令的类别。ISO/IEC 7816-3 中规定，值‘FF’是无效的。最高位 b8 是用来区别通用 CLA 和专用 CLA 的。b8 置 0，表示通用 CLA。000x xxxx 和 01xx xxxx 将在下面介绍，001x xxxx 保留作将来使用。表 8-3 规定了 CLA＝000x xxxx 时各位的含义。其中，b8，b7，b6 置为 000；b5 控制命令链；b4 和 b3 指示安全报文传输；b2 和 b1 编码从 0 到 3 的逻辑通道号。

表 8-3　CLA＝000x xxxx 格式

b8	b7	b6	b5	b4　b3	b2　b1	含　义
0	0	0	x	——	——	命令链控制：
0	0	0	0	——	——	• 本命令是命令链的最后一条或命令链仅此一条命令
0	0	0	1	——	——	• 本命令不是命令链的最后一条
0	0	0	—	x　x	——	安全报文传输 SM 指示：
0	0	0	—	0　0	——	• 无 SM 或无指示
0	0	0	—	0　1	——	• 专用 SM 格式
0	0	0	—	1　0	——	• 命令头不参与鉴别
0	0	0	—	1　1	——	• 命令头参与鉴别
0	0	0	—	——	x　x	从 0 到 3 的逻辑通道号

表 8-4 规定了 CLA＝01xx xxxx 时各位的含义。其中，b8，b7 置为 01；b6 指示安全报文传输；b5 控制命令链；b4 到 b1 编码从 0 到 15，该值加上 4 即为从 4 到 19 的逻辑通道号。

表 8-4　CLA＝01xx xxxx 格式

b8	b7	b6	b5	b4	b3	b2	b1	含　义
0	1	x	—	—	—	—	—	安全报文传输 SM 指示：
0	1	0	—	—	—	—	—	• 无 SM 或无指示
0	1	1	—	—	—	—	—	• 命令头不参与鉴别
0	1	—	x	—	—	—	—	命令链控制：
0	1	—	0	—	—	—	—	• 本命令是命令链的最后一条或命令链仅此一条命令
0	1	—	1	—	—	—	—	• 本命令不是命令链的最后一条
0	1	—	—	x	x	x	x	从 4 到 19 的逻辑通道号

3. 指令字节 INS

INS 指明要操作的命令。根据 ISO/IEC 7816-3 的规定，值‘6X’和‘9X’是无效的。表 8-5 列出了 ISO/IEC 7816 中规定的命令和其隶属的标准。

表 8-5　ISO/IEC 7816 中规定的命令

序号	命 令 名 称	INS	隶 属 标 准
1	Create File	E0	7816-9
2	Select File	A4	7816-4
3	Manage Channel	70	7816-4
4	Delete File	E4	7816-9
5	Deactivate File	04	7816-9
6	Activate File	44	7816-9
7	Terminate DF	E6	7816-9
8	Terminate EF	E8	7816-9
9	Terminate Card Usage	FE	7816-9
10	Read Binary	B0，B1	7816-4
11	Write Binary	D0，D1	7816-4
12	Update Binary	D6，D7	7816-4
13	Search Binary	A0，A1	7816-4
14	Erase Binary	0E，0F	7816-4
15	Read Records	B2，B3	7816-4
16	Write Record	D2	7816-4
17	Update Record	DC，DD	7816-4
18	Append Record	E2	7816-4
19	Search Record	A2	7816-4
20	Erase Record	0C	7816-4
21	Get Data	CA，CB	7816-4
22	Put Data	DA，DB	7816-4
23	Internal Authenticate	88	7816-4

续表

序号	命 令 名 称	INS	隶 属 标 准
24	Get Challenge	84	7816-4
25	External Authenticate	82	7816-4
26	General Authenticate	86，87	7816-4
27	Verify	20，21	7816-4
28	Change Reference Data	24	7816-4
29	Enable Verification Requirement	28	7816-4
30	Disable Verification Requirement	26	7816-4
31	Reset Retry Counter	2C	7816-4
32	Manage Security Environment	22	7816-4
33	Perform Security Operation	2A	7816-8
34	Generate Public Key Pair	46	7816-8
35	Get Response	C0	7816-4
36	Envelope	C2，C3	7816-4
37	Perform Scql Operation	10	7816-7
38	Perform Transaction Operation	12	7816-7
39	Perform User Operation	14	7816-7

4. 状态字节 SW1-SW2

SW1-SW2 指示了处理状态。在 ISO/IEC 7816-3 中规定，所有不同于‘6XXX’和‘9XXX’的值都是无效的，‘60XX’也是无效的。

值‘61XX’、‘62XX’、‘63XX’、‘64XX’、‘65XX’、‘66XX’、‘68XX’、‘69XX’、‘6AXX’、‘6CXX’、‘6700’、‘6B00’、‘6D00’、‘6E00’、‘6F00’、‘9000’在 ISO/IEC 7816-4 中做了相应的定义，是通用状态字。而‘67XX’、‘6BXX’、‘6DXX’、‘6EXX’、‘6FXX’(XX 不为 00)和‘9XXX’(XXX 不为 000)都是专用状态字。

图 8-12 所示为 SW1-SW2 的值为‘9000’和从‘61XX’到‘6FXX’的结构图。

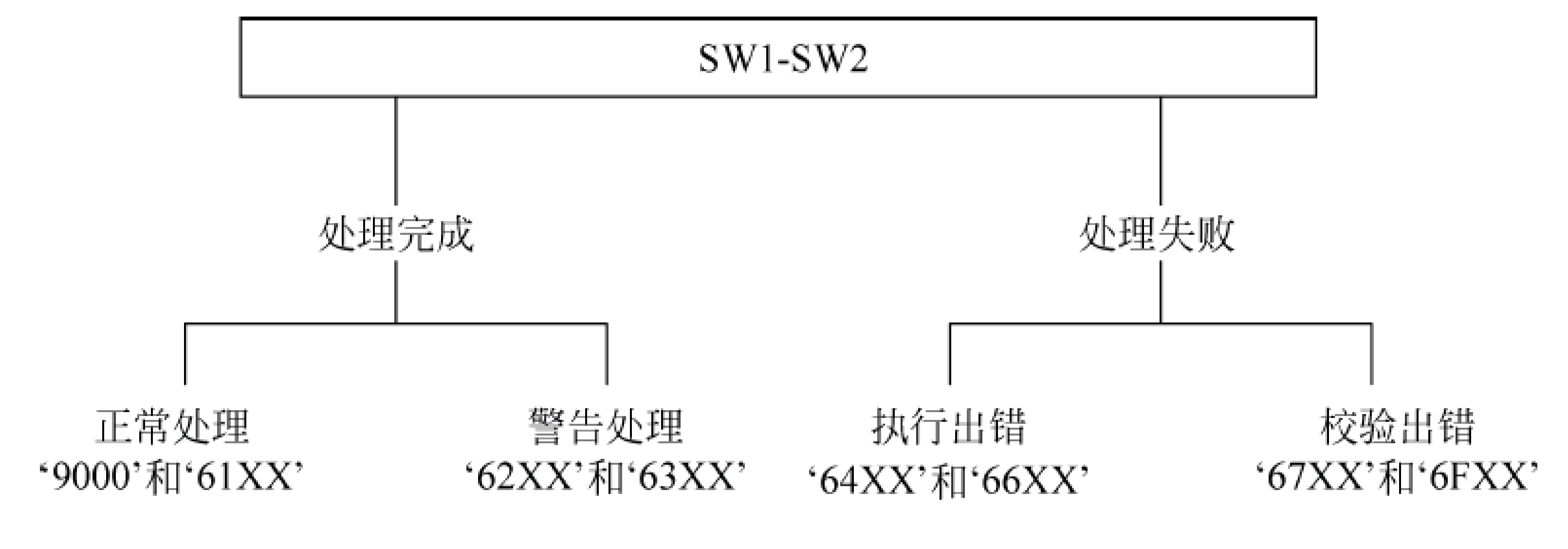


图 8-12 SW1-SW2 取值结构图

8.2 接触式读写设备设计

智能卡根据通信协议可分为两种：接触式和非接触式，对应的读写机具也分为接触式读写机具和非接触式读写机具。

接触式智能卡采用的传输模式如表 8-6 所示，目前最常用的是 T=0 和 T=1 两种，无论是采用 T=0 协议还是 T=1 协议，智能卡在信息交换时使用的都是异步通信模式；而且由于智能卡的数据端口只有一个，因此信息交换也只能采用半双工的方式，即在任一时刻，数据端口上最多只能有一方(智能卡或者读/写设备)在发送数据。T=0、T=1 协议的不同之处在于它们数据传输的单位和格式不一样：T=0 协议以单字节的字符为基本单位，T=1 协议则以有一定长度的数据块为传输的基本单位。

表 8-6 在 ISO/IEC 国际标准中定义或保留的接触式卡传输协议

协议类型	协议名称	说明
T=0	异步半双工字符传输协议	目前正在使用
T=1	异步半双工块传输协议	目前正在使用
T=2、3	保留用于将来的全双式通信	
T=4	留用于加强异步半双工字符传输协议	
T=5、13	保留	
T=14	保留用于非 ISO 标准协议	目前正在应用
T=15	保留用于将来扩展	

8.2.1 T=0 传输协议

T=0 传输协议是第一个形成国际规范的智能卡传输协议，它的特点是应用最广、存储区使用最小，并且协议设计简单。目前这项协议被广泛地应用于手机 SIM 卡中。T=0 传输协议在 ISO/IEC 7816-3 中被标准化。

T=0 传输协议是面向字节的。传输的数据单元包括用 CLA、INS、P1 和 P2 以及 1 个可变长度的条件体来表示。

接触式智能卡数据在 I/O 上以图 8-13 所示的字符帧方式传输。

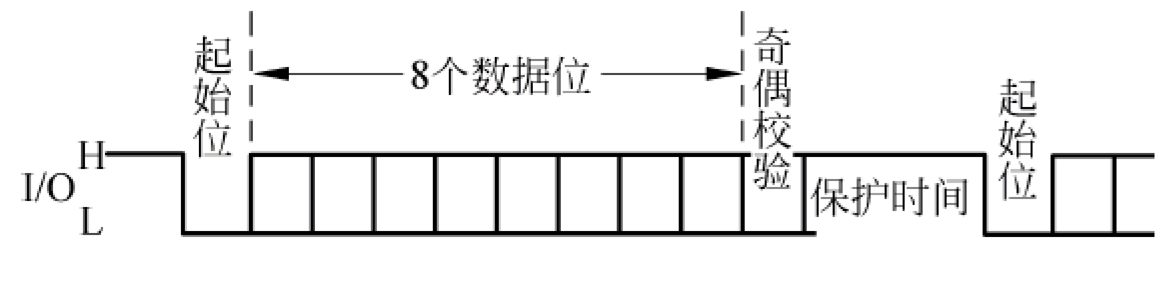


图 8-13 字符帧传输方式

起始位由接收端通过对 I/O 周期采样获得，采样周期应小于 0.2etu。两个连续字符起始位上升沿之间的间隔时间等于(10±0.2)etu 加上 1 个保护时间(最少两个 etu)。在保护时间内，卡与终端都应处于接收模式(I/O 为高电平状态)。如果卡或终端作为接收方检测出奇偶错误，则 I/O 被置为低电平，以此向发送方表明出现错误。

在 EMV2000-Book1 中，给出了正常流程下的 CASE1～CASE4 的 T=0 传输协议处理流程，如图 8-14 所示。通过这个图可以加深对 T=0 传输协议的理解，更多细节可参见 ISO/IEC 7816 和 EMV 规范。

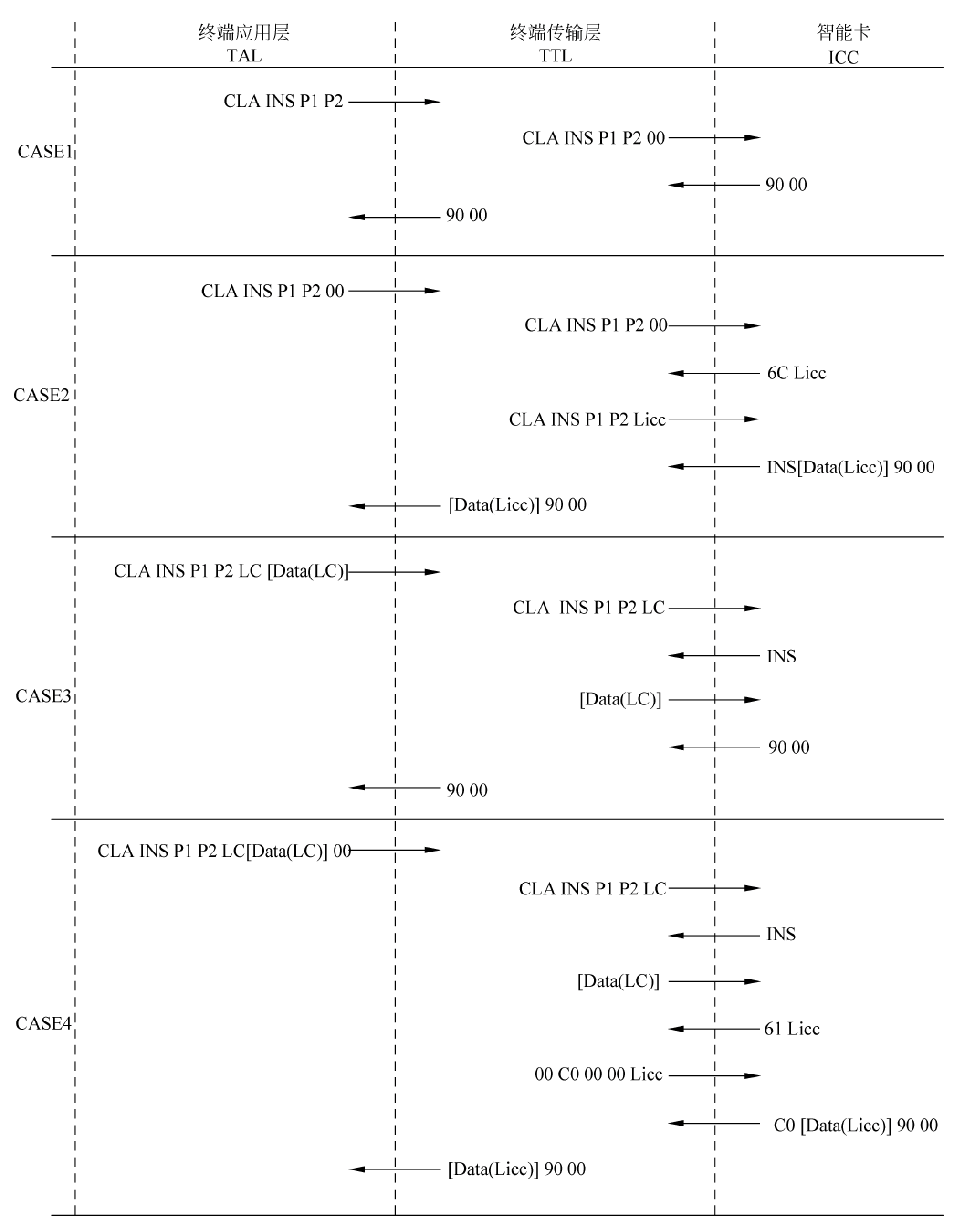


图 8-14　T=0 传输协议流程

8.2.2　T=1 传输协议

符合 T=1 传输协议的智能卡在复位之后，与读写设备之间以分组的形式进行数据传输，每一个数据分组称为一个数据帧。数据帧由开始字段(包括结点地址、协议控制字节和长度字节，该字段必备)、信息字段 INF(可选的)和结尾校验字段 EDC(必备)组成，具体的

数据帧的格式如图 8-15 所示。

结点地址 NAD	协议控制字节 PCB	长度 LEN	信息字段 INF	校验字节 EDC
1字节	1字节	1字节	ELN字节 (0-254)	LRC:1字节 CRC:2字节

图 8-15　T=1 协议数据帧格式

其中,PCB 包含控制数据传输所需的信息,定义了 3 种基本分组类型:

(1) I-block。包含应用层所用的信息。

(2) R-block。用来传输正确(ACK)的或者出错(NAK)的确认信息。不含 INF 字段。它表示对最后一次接收到的数据块的确认信息。

(3) S-block。用来在读写设备和卡之间交换控制信息。

对于符合 T=1 协议的卡,终端将应用层(terminal application layer,TAL)传送的 APDU 封装成 BLOCK,然后发送给卡,卡回传应答数据组。特别注意的是 PCB 需要设置顺序号。更多的细节可以参考 ISO/IEC 7816 标准和 EMV 2000 标准。在接触式卡片中 T=1 传输协议用得较少,主要用于接触式 Java 卡。T=1 传输协议经过扩展和改造,形成 T=CL 的非接触式传输协议,参见本章的第 8.3 节。

8.2.3 整体结构

手持式接触式读卡器系统结构见图 8-16。

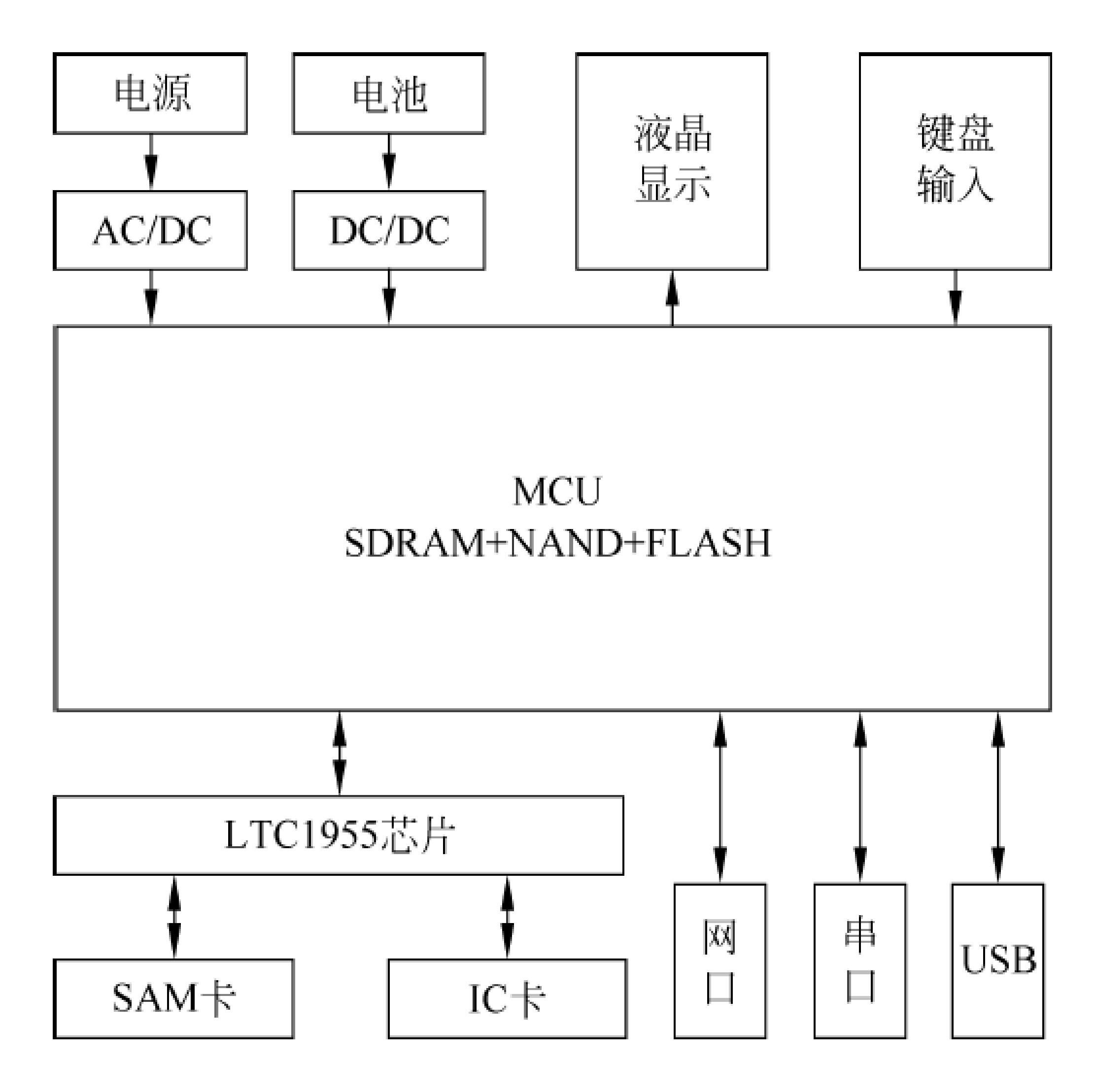


图 8-16　接触式读卡器系统结构

该读卡器具有如下功能:

(1) 读卡器功能。两个 IC 卡槽,一个用户卡,一个 SAM 卡。接口芯片采用 LINEAR 公司的 LTC1955,同时支持用户卡和 SAM 卡,支持 1.8/3V/5V 卡,外围电路简单,所需 I/O

口少。由于 LTC1955 的供电能力只有 50mA，所以为了能够读取大电流的 IC 卡芯片，另外增加一路 3.3V，260mA 的电源给 IC 卡单独供电。

(2) 通信接口和开关键。本系统具有 USB 接口、串口和网络接口。具有少量开关键、简单输入和电源控制等。

(3) PDA 功能。在大屏幕图形 LCD 上实现图形界面，同时具有网络浏览器、文本编辑、图片浏览等功能。

(4) 电源管理功能。本系统选择锂电池，体积小，重量轻，是 PDA 产品的首选电池，然后根据锂电池的特性选择相应的 DC-DC 变换电路。该系统是手持终端，因此对所用模块或芯片都应保留关断功能，在空闲时关断，以节省电池。

(5) 可扩展功能。如增加指纹识别功能，可采用 SW6888 芯片实现指纹识别模块。

8.2.4 接口芯片

在接触式读写设备中，智能卡读写接口芯片有很多种，其中 LINEAR 公司的 LTC1955 芯片性价比较高。LTC1955 双智能卡接口可以为两块智能卡提供全部所需的电源管理、控制、静电释放和故障检测电路。采用一个倍压电荷泵和两个低压降线性调整器，LTC1955 从一个 2.7～5.5V 输入产生两路独立的 5V、3V 或 1.8V 电压。两通道都有支持 EMV 和 ISO/IEC 7816 智能卡标准所需的引脚，其中一个通道有额外的控制引脚(智能卡接触凸点 C4 和 C8)以支持现有的存储卡。整个芯片通过串行接口由微控制器来控制。接口电路参见图 8-17 所示。

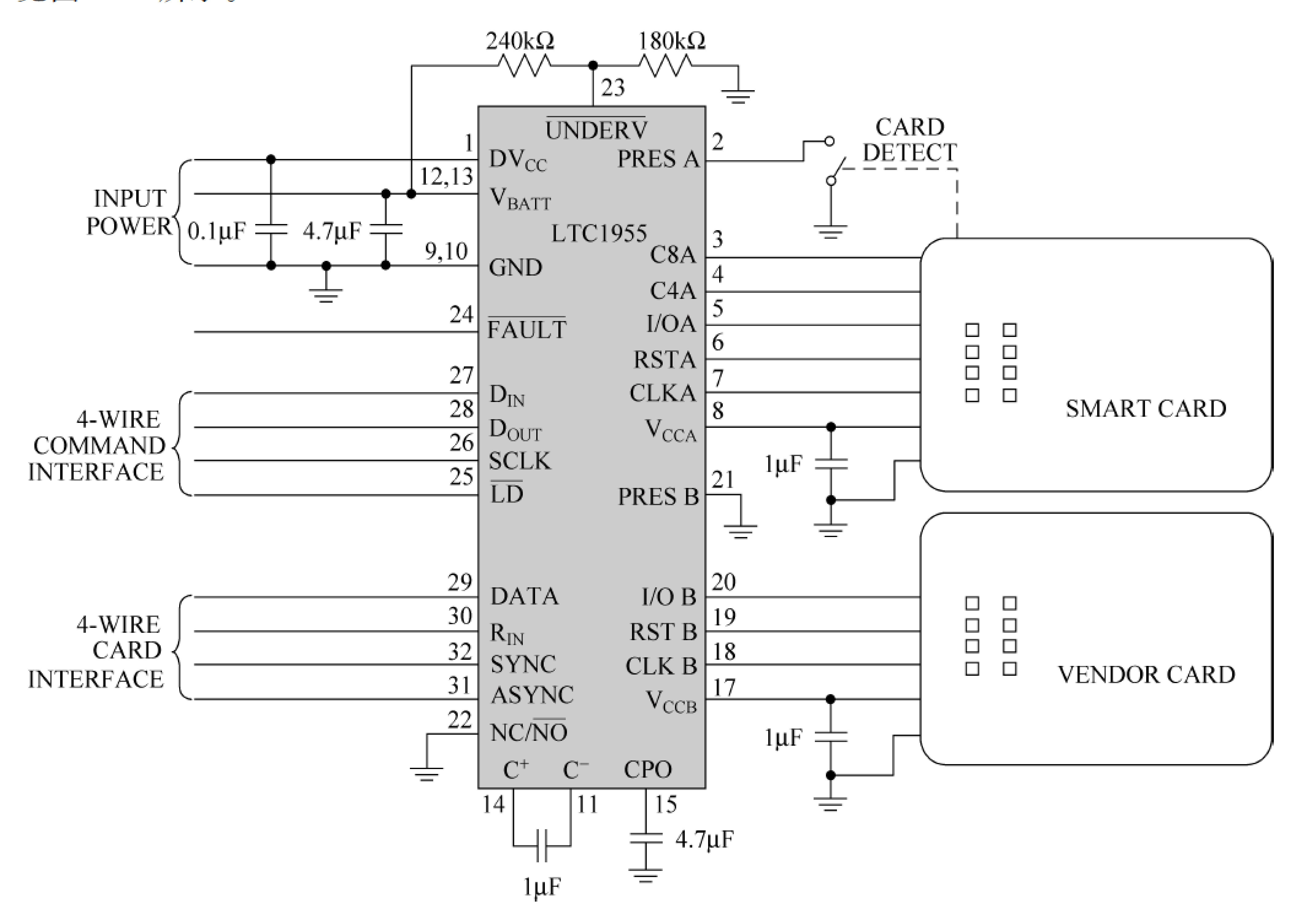


图 8-17 LTC1955F 芯片接口电路

8.2.5 时序和流程

所谓接触式通信接口是指智能卡通过金属管脚直接与读写设备相连，由此为智能卡提供电源、复位信号、时钟信号以及数据输入输出。一般说来，接触式智能卡正常用卡过程可划分为如下几个阶段：

(1) 把智能卡插入读写设备，接通各触点。

(2) 读写设备给卡发复位信号，并在读写终端和卡之间建立通信。

(3) 执行交易。

(4) 释放触点并取出智能卡。所谓释放触点，即使读写设备将其各触点去电，持卡人可拔出智能卡结束用卡过程。

交易过程中，数据以异步半双工方式经 I/O 线在终端和卡间双向传送。由终端向卡提供时钟信号，并以此来控制交易的时序。信息交换时的数位和字符的规定符合 ISO/IEC 7816 标准。

机具与智能卡之间是串行通信，所以时序的严格准确是非常重要的。时序主要分为智能卡复位时序、正常通信时序。下面介绍智能卡复位过程的时序。智能卡冷复位的时序如图 8-18 所示，读写机具在 T_0 时刻将时钟信号输出给智能卡，此后的 200 个时钟周期内，智能卡和读写机具都处于接收状态。智能卡在收到时钟信号后，进行内部复位，然后 400～40 000 个时钟周期内，卡会发出一个 ATR(answer to reset)响应。当机具将时钟信号提供给卡后，要求 RST 至少在 40 000 个时钟周期(图中 T_0～T_1 时段)内保持为低电平，在这段时间内如果智能卡仍然没有发送 ATR 信号，接口设备再将 RST 信号置为高电平。按照协议，智能卡在 RST 上升沿后的 t_1 时间(400～40 000 个时钟周期)内必须将 ATR 数据发送到 I/O 线上。如果 RST 上升沿后再经过 40 000 个时钟周期后接口设备仍然没有收到卡发出的 ATR 数据，表示接收 ATR 数据超时，接口设备就会把 RST 置为低电平，并使智能卡处于无效状态。

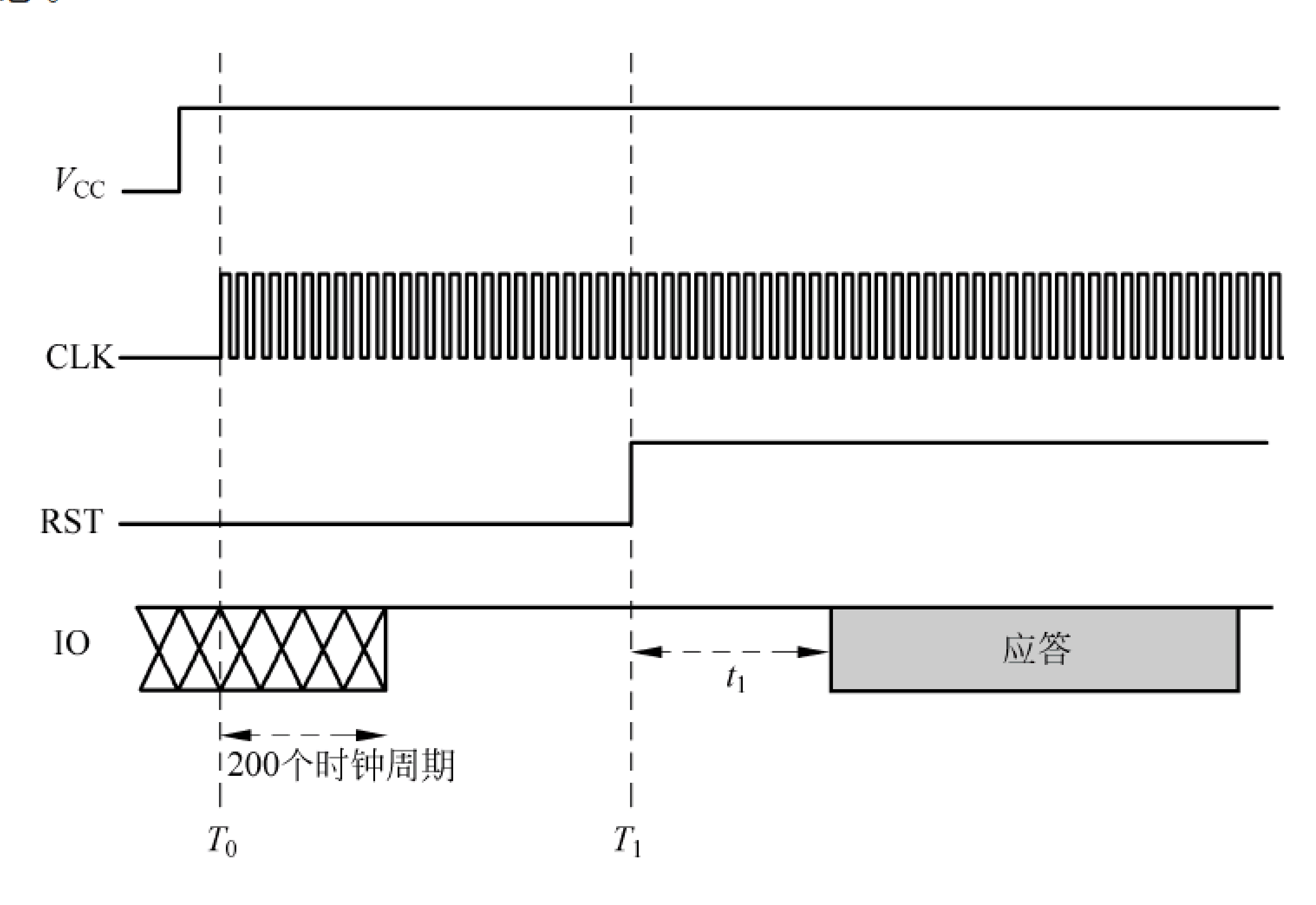


图 8-18 智能卡冷复位时序

I/O 线上所用的数位宽度被确定为基本时间单位(elementary time unit,ETU),它和时钟频率间存在着线性关系。数据在 I/O 线上以字符帧传送,所用约定在 IC 卡复位应答传送的起始字符 TS 中予以规定。

在初始复位状态下,当机具给智能卡提供电源信号时,它们都处于接收状态,等待接收 I/O 数据线上的数据,一旦复位操作完成,智能卡处于发送模式,接口设备仍处于接收状态,于是智能卡发送 ATR 数据给接口设备。当接口设备收到 ATR 数据后,机具改为发送模式,智能卡处于接收模式,这样就可以向智能卡发出各种命令让智能卡执行,之后,机具和智能卡再分别处于接收、发送模式进行数据传输。时钟线和数据 I/O 线都需要具备支持一定范围内的数据传输速度的能力,具体使用的数据传输速度是由智能卡发向接口设备的 ATR 里面的数据确定的。数据传输速度由 I/O 数据线上一个比特的持续时间定义,这个时间被称为基本时间单位(ETU)。在 ATR 数据传输过程中的 ETU 一般为 372/fi,fi 是时钟频率,它的范围为 1～5MHz。

机具端读写 T=0 卡片的流程如图 8-19 所示。

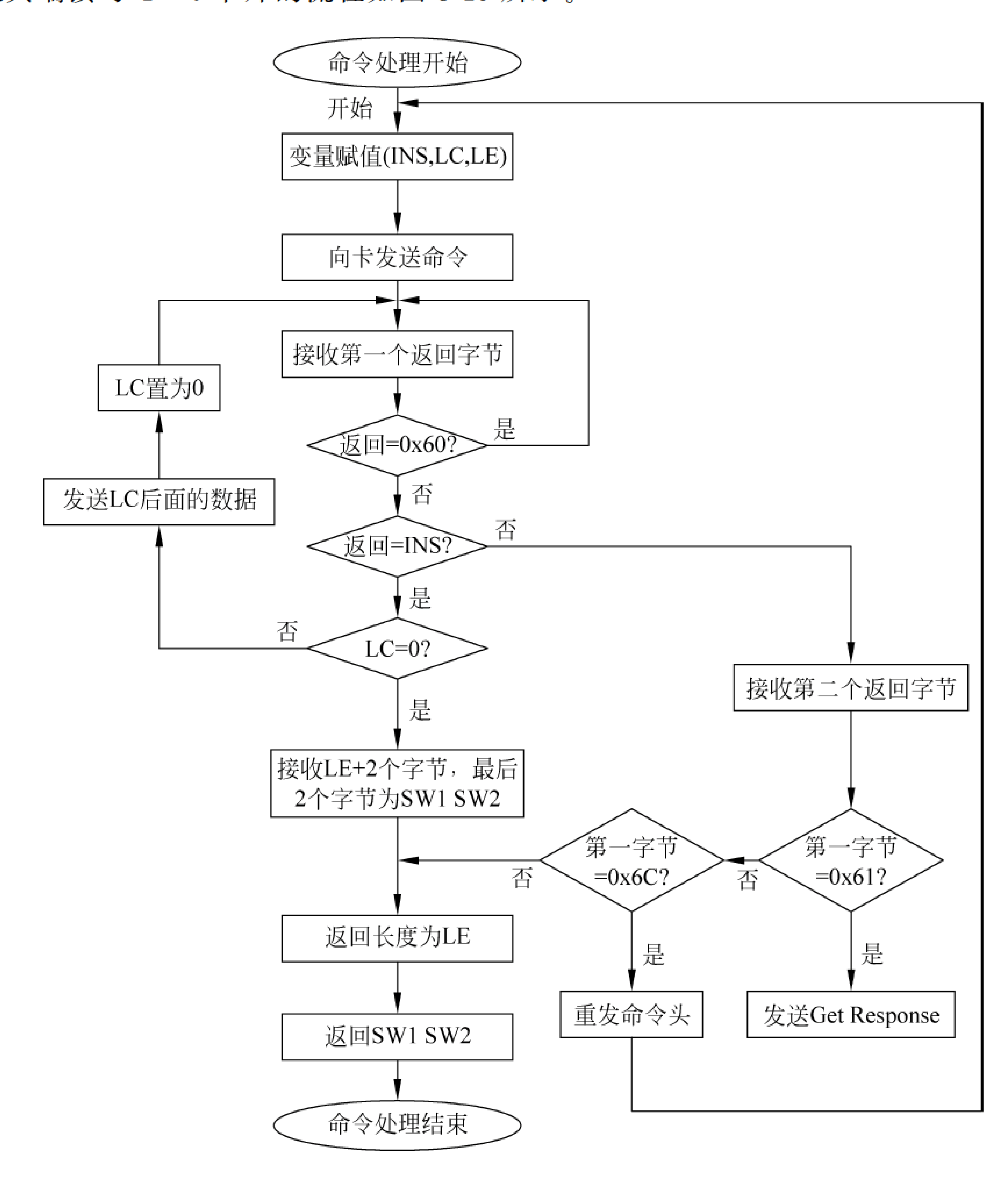


图 8-19　T=0 方式处理流程

8.3 非接触式读写设备设计

非接触式IC卡的集成电路不向外引出触点，而是通过射频收发电路和天线与读写器实现信息交换。读写器发送给卡的射频信号一方面充当数据信息的载体，同时它也为非接触式IC卡提供必需的电源。非接触式IC卡由于具有与读写器无机械接触，而是借助电磁波进行通信等条码卡、磁卡和接触式智能卡无法比拟的优点，使之一经问世，便立刻引起了广泛的关注，并以惊人的速度得到推广应用。

非接触式IC卡因作用距离的不同而分为3个不同的国际标准。

(1) 非接触式密耦合卡(close coupled IC card，CICC)，对应标准ISO/IEC 10536-1～4。

(2) 非接触式近耦合卡(proximity IC card，PICC)，对应标准ISO/IEC 14443-1～4。

(3) 非接触式疏耦合卡(vicinity IC card，VICC)，对应标准ISO/IEC 15693-1～3。

非接触式近耦合卡的作用距离在10cm左右，目前绝大部分的民用系统都采用的是非接触式近耦合卡，这种卡采用的国际标准ISO/IEC 14443分为4个部分：

(1) 物理特性的介绍。ISO/IEC 14443-1描述了非接触式IC卡的尺寸应与国际标准ISO/IEC 7816中的规定相符，除此以外，还应满足在紫外线、X射线、交流电场、静电、静磁场、工作温度、动态弯曲和动态扭曲等方面提出的要求。

(2) 关于射频能量和信号接口的介绍。ISO/IEC 14443-2描述了在近耦合设备(PCD)和近耦合卡(PICC)之间提供能量和半双工通信的射频信号特征。PCD产生耦合到PICC的RF电磁场，用以传送能量并被调制来用于通信。射频工作频率是13.56MHz±7kHz。PCD设备产生的磁场强度在可操作距离范围内任何地方都不应小于最小场强和大于最大场强，在此范围内PICC应能不间断地工作。从智能卡与读写器之间的通信方式来讲，符合国际标准ISO/IEC 14443的非接触式智能卡目前主要有两类：Type A和Type B。两类标准主要差别在于调制、编码和反碰撞方式的不同。Type A采用动态二进制搜索法实现多张卡片的访问，完成反碰撞功能。Type B采用时隙ALOHA法实现多张卡片的分时隙进行访问，完成反碰撞功能。

(3) 关于初始化和防冲撞的介绍。ISO/IEC 14443-3描述了在一个PCD设备的射频区域内查询PICC卡字节和帧的格式，初始化请求命令和请求命令的响应的内容，在几张卡中对一张卡的检测和通信的方法(即防冲突)和其他一些需要初始化的参数。该部分定义了帧的格式和时序，PCD或PICC发送的帧都包含一个帧头、一个帧尾和中间的数据信息或错误检测位。PCD与PICC之间采用半双工的通信方式，帧与帧之间的延迟时间也在标准里作了具体的规定。

(4) 关于传输协议的介绍。ISO/IEC 14443-4规定了一个非接触式环境中所必需的半双工的块传输协议，并定义了激活的和没有激活的协议的流程。

ISO/IEC 14443规范和ISO/IEC 7816规范之间的对应关系如图8-20所示。

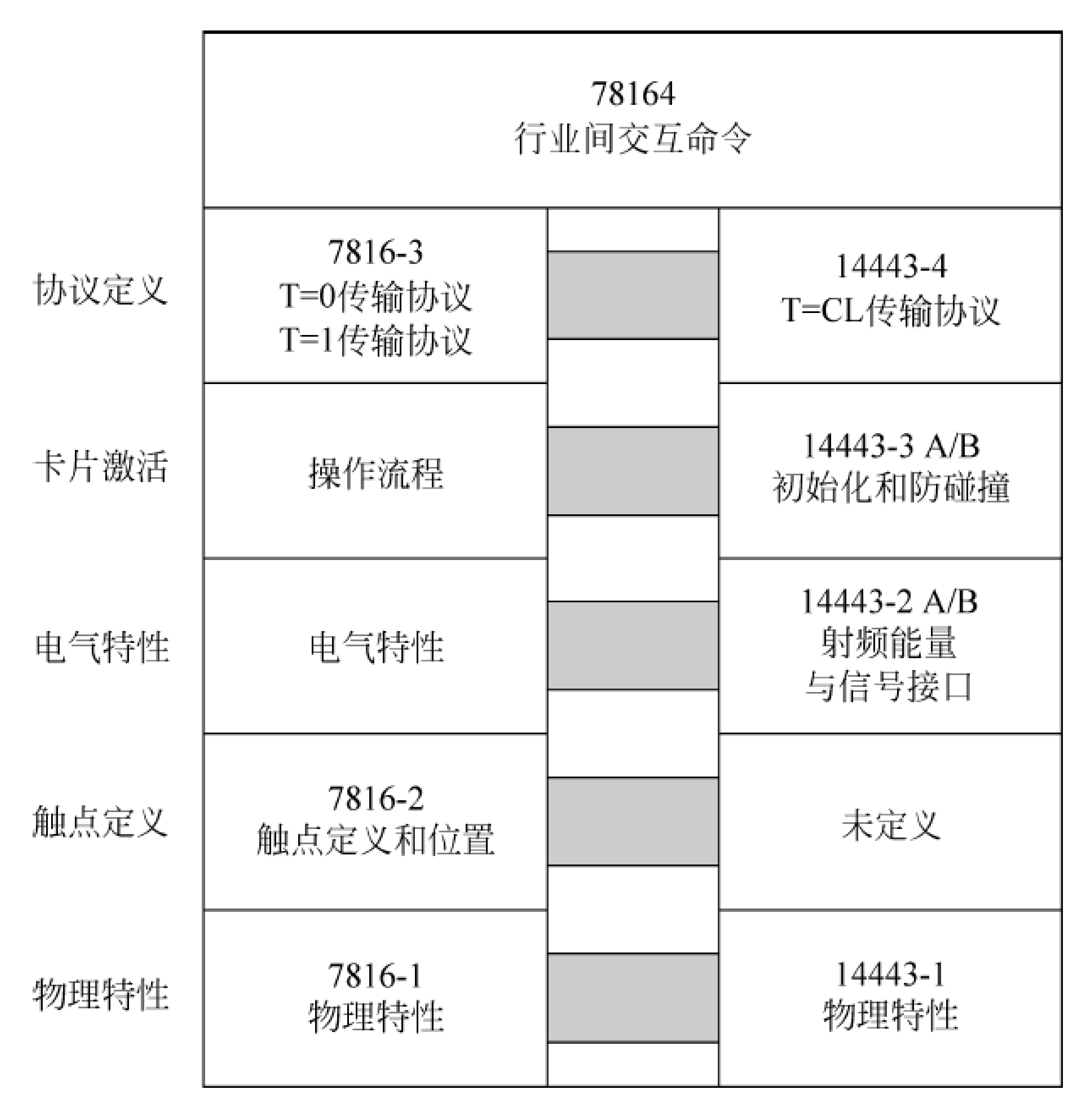


图 8-20 ISO/IEC 7816 和 ISO/IEC 14443

8.3.1 Type A 协议

ISO/IEC 14443 中 Type A 是由 NXP 等半导体公司最先开发和使用的。Type A 技术是一个非常优秀的非接触技术，设计简单扼要，应用项目的开发周期可以很短，同时又能起到足够的保密作用，可以适用于非常多的应用场合。在亚洲等地区，Type A 技术和产品占据了很大的市场份额。

代表 Type A 非接触式智能卡芯片主要有 Mifare_Light（MF1 IC L10 系列）、MIFARE1（S50 系列、内置 ASIC）、Mifare2（即 Mifare Pro）、MF2 ICD8x 系列（接触/非接触双接口系列、内置兼容 Intel 18051 的微处理控制器 MCU 等）。相应的 Type A 卡片读写设备核心 ASIC 芯片，以及由此组成的核心保密模块 MCM（Mifare_Core_module）的主要代表有：RC150、RC170、RC500 等，以及 MCM200、MCM500 等。

图 8-21 为从机具 PCD 角度观察的 Type A 协议流程。

读写机具向卡传送信号，通过 13.65MHz 的射频载波传送信号。其采用的方案为同步、改进的 Miller 编码方式，通过 100%ASK 传送；当卡向读写机具传送信号时，通过调制载波传送信号。使用 847kHz 的副载波传送 Manchester 编码。简单说，当表示信息 1 时，信号会有 0.3 微妙的间隙，当表示信息 0 时，信号可能有间隙也可能没有，这与前后的信息有关。这种方式的优点在于信息区别明显，受干扰的机会少，反应速度快，不容易误操作；缺点是在需要持续不断地提高能量到非接触式卡时，能量有可能会出现波动。

在机具和非接触式智能卡之间采用串行命令，如图 8-22 所示。命令格式的简单解释如下，更加详细的内容可参见 ISO/IEC 14443-3/4 规范。

（1）REQA/WUPA 命令

REQA 和 WUPA 命令属于短帧格式，短帧格式分为 1 个起始位、7 个数据位（低位在前）和 1 个结束位组成。REQA/WUPA 命令格式如表 8-7 所示。

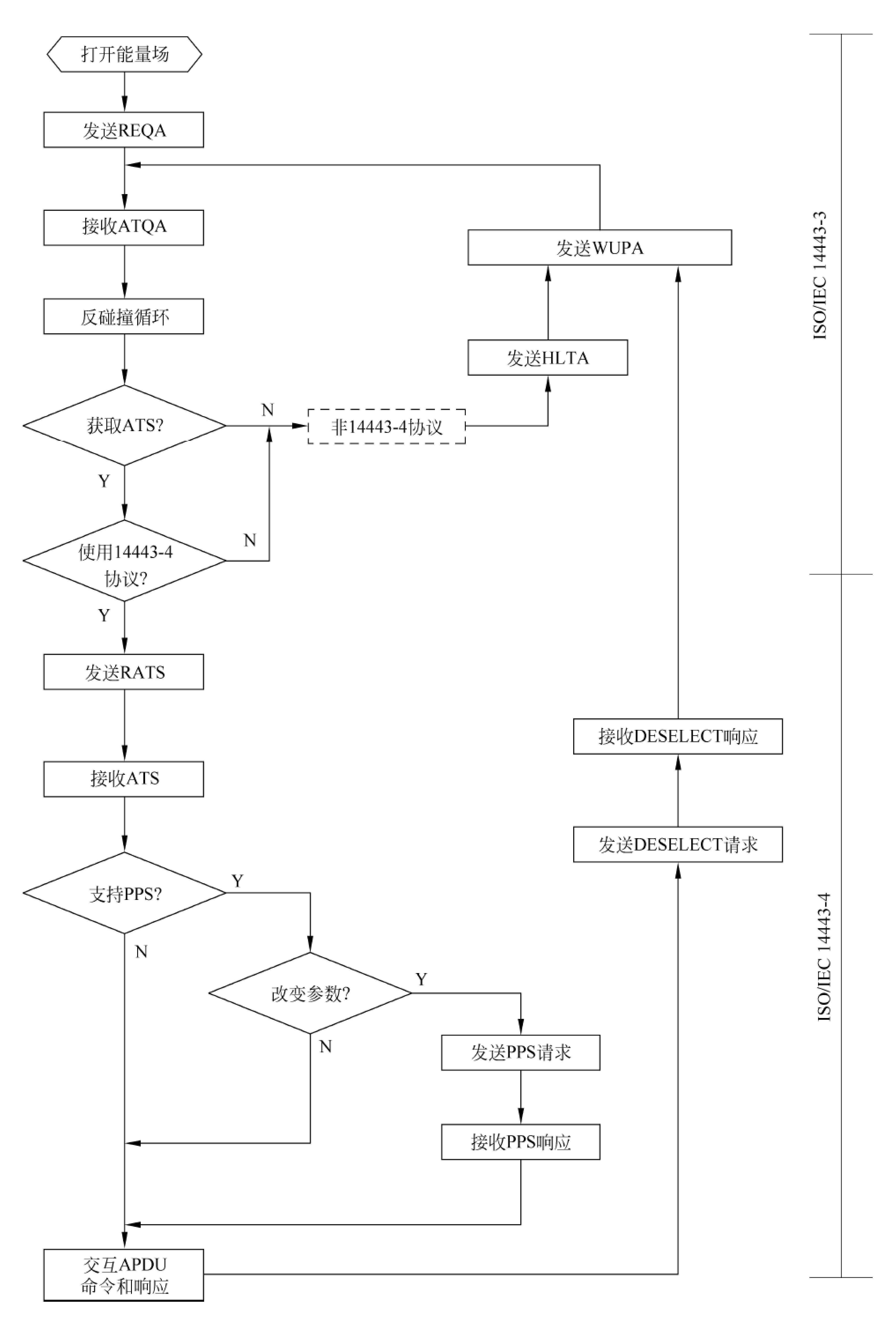


图 8-21 Type A 协议的机具端处理流程

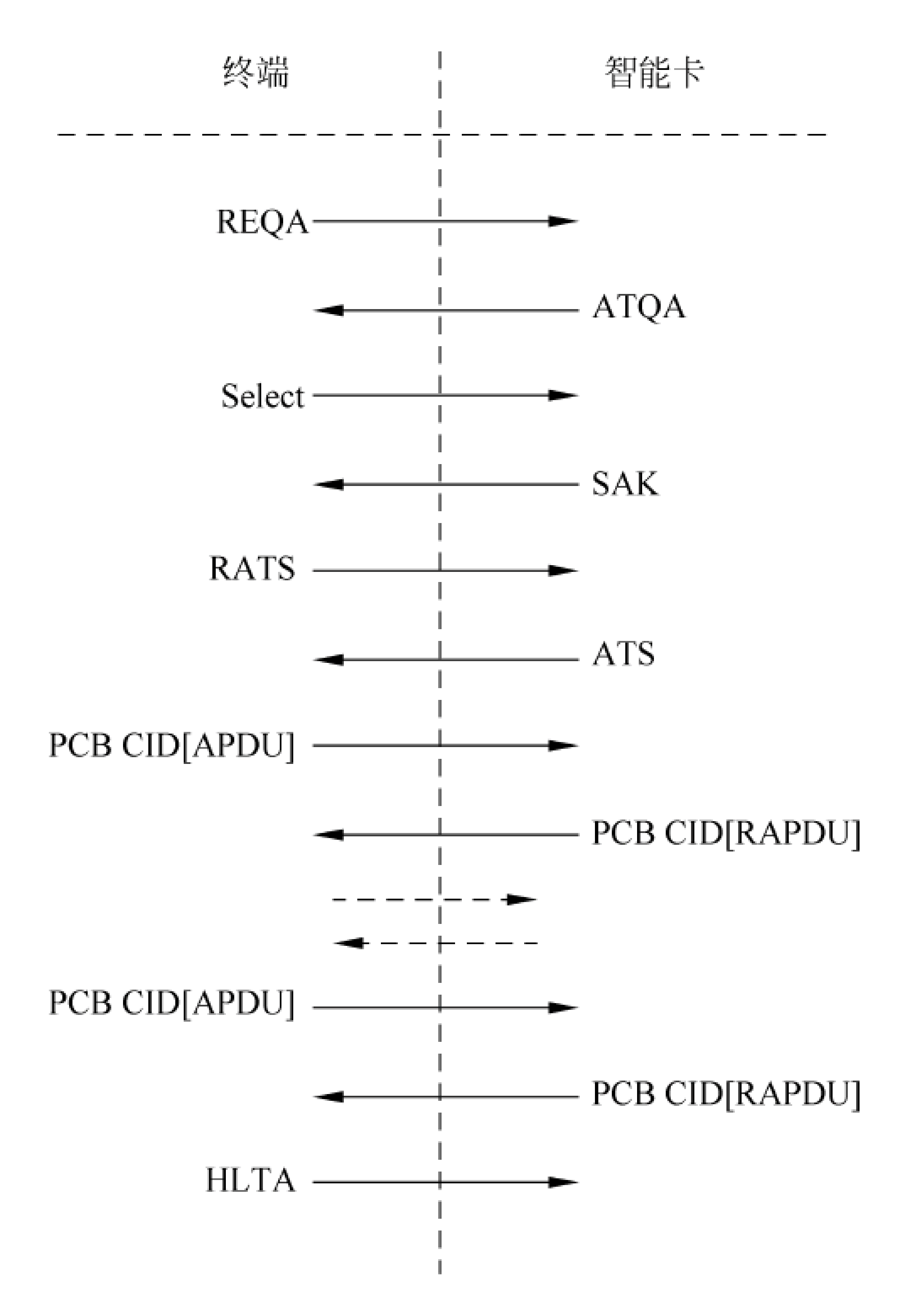


图 8-22　TypeA 协议中正常流程的命令和响应

表 8-7　REQA/WUPA 命令

b7	b6	b5	b4	b3	b2	b1	含　　义
0	1	0	0	1	1	0	‘26’＝REQA
1	0	1	0	0	1	0	‘52’＝WUPA
0	1	1	0	1	0	1	‘35’＝可选时隙方法
1	0	0	X	X	X	X	‘40’-‘4F’＝专用
1	1	1	1	X	X	X	‘78’-‘7F’＝专用
其他							RFU

后面的命令都遵守标准帧格式，标准帧格式用于数据交换，包括 1 个起始位、N*（8 个数据位＋1 个校验位）和 1 个结束位组合而成。

（2）ATQA 响应

PICC 接收到 REQA 或 WUPA 命令后，返回 ATQA 响应。ATQA 响应为两字节长度，信息结构如表 8-8 所示。

表 8-8　ATQA 响应编码格式

	ATQA 高字节								ATQA 低字节							
	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
ISO/IEC 14443-3 规范中位编码定义	RFU				专用编码				UID 大小比特帧		RFU	比特帧反碰撞				

续表

	ATQA 高字节					ATQA 低字节							
支持 212kbps 通信速率					1								
支持 424kbps 通信速率				1									
支持 848kbps 通信速率			1										
单个 UID						0	0						
两个 UID						0	1						
三个 UID						1	0						
RFU						1	1						
比特位反碰撞									1	0	0	0	0
比特位反碰撞									0	1	0	0	0
比特位反碰撞									0	0	1	0	0
比特位反碰撞									0	0	0	1	0
比特位反碰撞									0	0	0	0	1

(3) Select 命令

多张 PICC 同时放置在机具上时，多卡通信发生碰撞。利用 Anticollsion 和 Select 进行反碰撞循环，命令格式如图 8-23 所示。

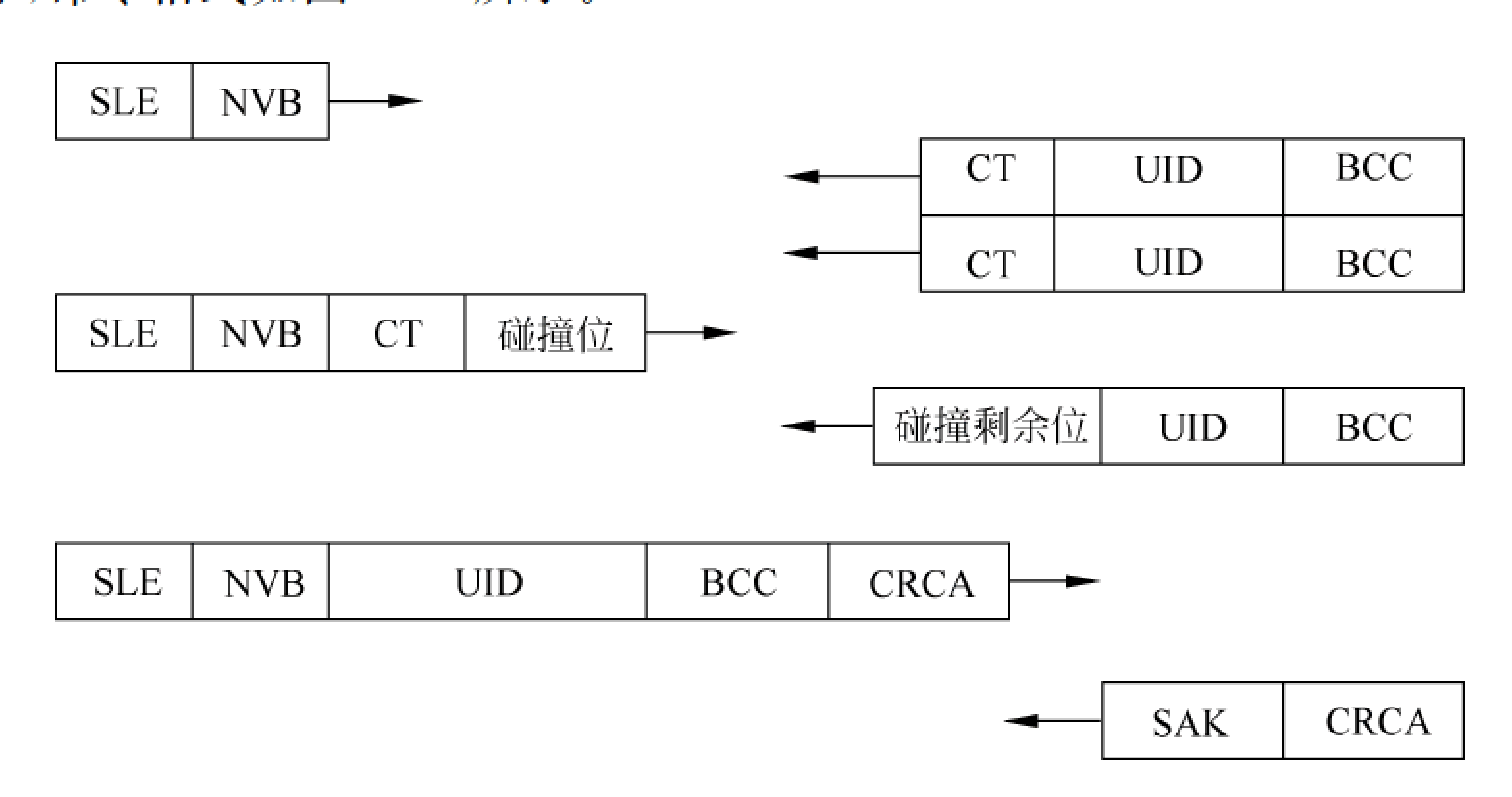


图 8-23　反碰撞循环示例

其中 SEL 字节编码如表 8-9 所示。

表 8-9　SEL 字节编码

b8	b7	b6	b5	b4	b3	b2	b1	说　　明
1	0	0	1	0	0	1	1	‘93’选择 UID CL1
1	0	0	1	0	1	0	1	‘95’选择 UID CL2
1	0	0	1	0	1	1	1	‘97’选择 UID CL3
1	0	0	1	其他				RFU

(4) SAK 响应

选择确认(select acknowlege,SAK)是 PICC 对 PCD 的 Select 命令的应答,是反碰撞流程的重要标志响应,用于通知 PCD:PICC 是否所有 UID CL 都已经经过比特帧反碰撞处理。SAK 采用如表 8-10 方式编码。

表 8-10 SAK 编码

含　　义	b8	B7	b6	b5	b4	b3	b2	b1
UID 不完整	RFU			RFU		1	RFU	
UID 完整						0		
PICC 符合 ISO/IEC 14443-4			1			0		
PICC 不符合 ISO/IEC 14443-4			0			0		

(5) RATS 命令

RATS(request for answer to select)命令将 PCD 的接收缓冲区大小 FSD 和对 PICC 的标识编号一起发给结束碰撞流程已被选择的卡片,请求其返回响应。RATS 命令格式如图 8-24 所示。

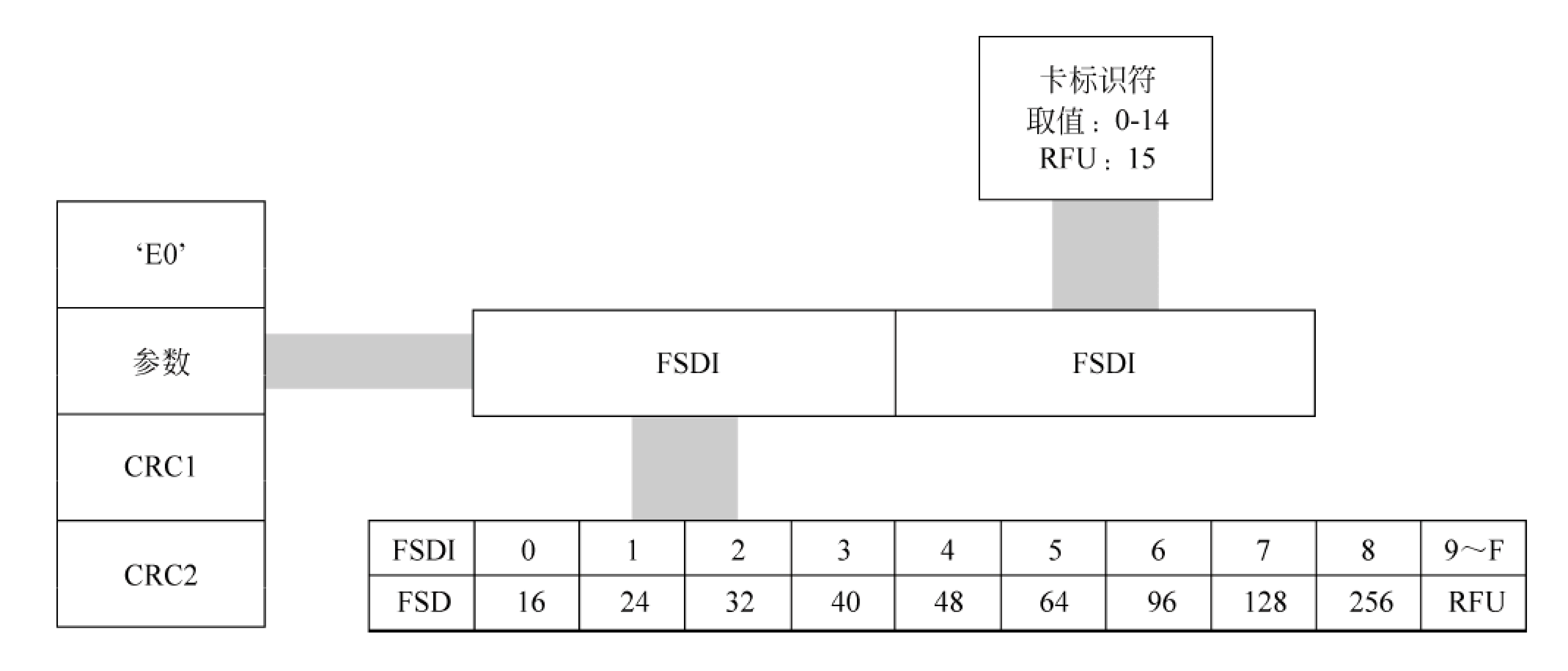


图 8-24　RATS 命令格式

(6) ATS 响应

PICC 通过 ATS 响应对 RATS 命令进行应答,表明该卡片已经被选中,并且将 PICC 的接口参数字节回送给 PCD。命令格式参见图 8-25 所示。

(7) HLTA 命令

PCD 采用 HLTA 命令使得 PICC 进入停止状态,不再进行任何通信,直至接收到 WUPA 命令,再次进入准备状态。PICC 对 HLTA 命令不再回发响应,直接进入停止状态。HLTA 命令格式如表 8-11 所示。

(8) 数据交换命令和响应

智能卡在激活之后,与读写设备之间以分组的形式进行数据传输,每一个数据分组被称为一个数据块或者数据帧。数据块由开始字段(必备)、信息字段 INF(可选的)和结尾校验字段 EDC(必备)组成,具体的数据帧的定义如图 8-26 所示。

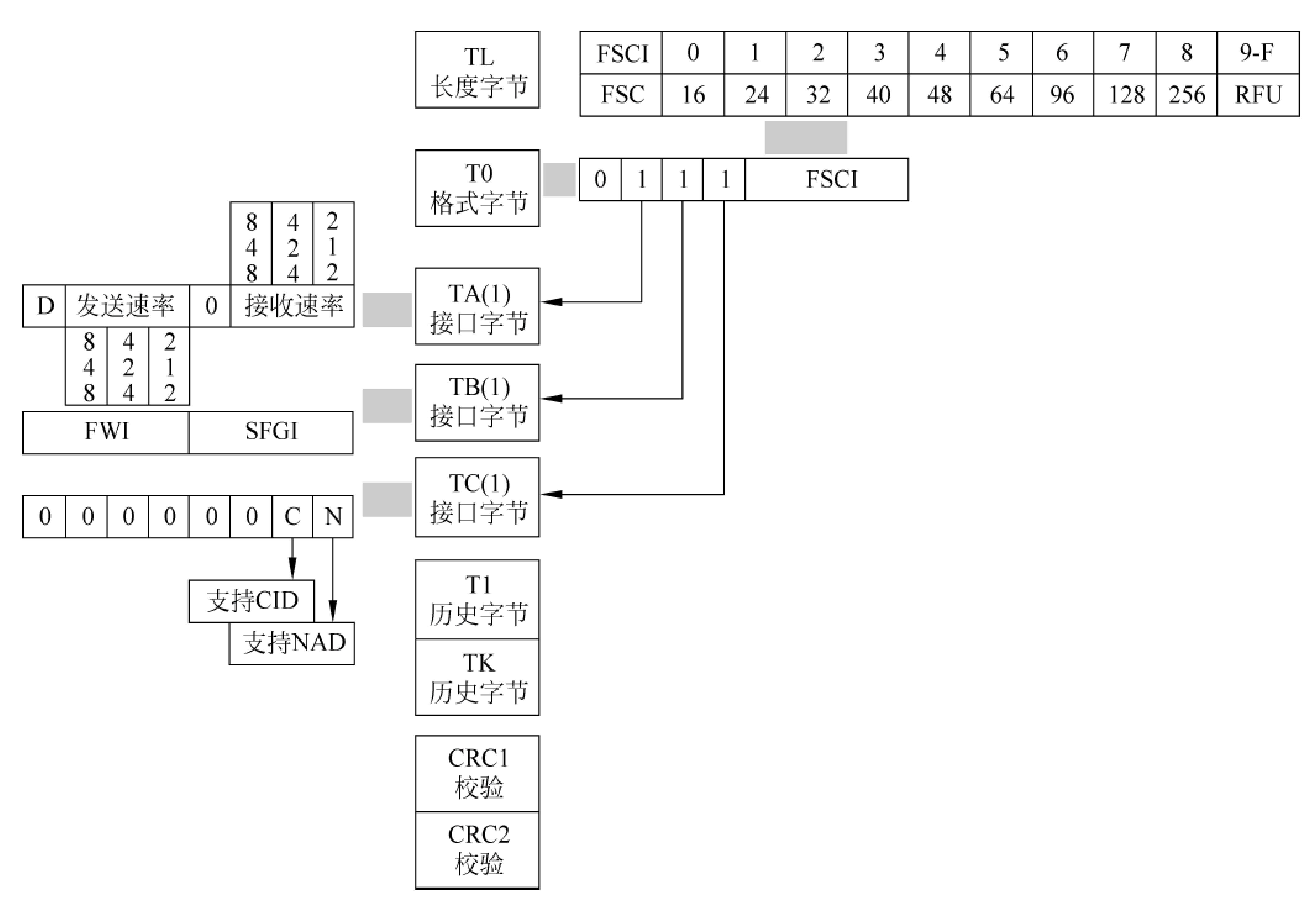


图 8-25　ATS 命令格式

表 8-11　HLTA 命令格式

‘50’	‘00’	CRC1	CRC2

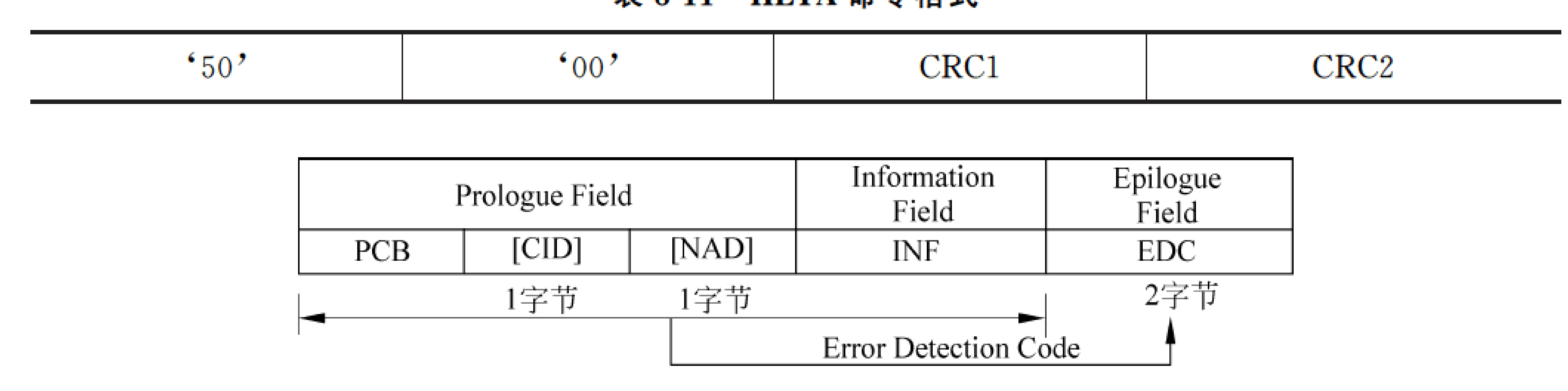


图 8-26　非接触式数据帧格式

PCB 包含控制数据传输所需的信息，定义了 3 种基本分组类型：

(1) I-block。包含应用层所用的信息。

(2) R-block。用来传输正确(ACK)的或者出错(NAK)的确认信息。不含 INF 字段。它表示对最后一次接收到的数据块的确认信息。

(3) S-block。用来在读写设备和卡之间交换控制信息。定义了两种 S-block，即用来申请改变 WTX(waiting time extension)的 S-block 和 DESELECT 的 S-block。前者包含一个字节 INF 字段，后者不包含 INF 字段。

8.3.2　Type B 协议

ISO/IEC 14443 中 Type B 是一个开放式的非接触式智能卡标准。所有的读写操作可以由具体的应用系统开发者定义。正因为这一点，它可以被世界上众多的智能卡厂家所广

泛接受。正是由于 Type B 具有开放式特点，所以每个厂家在具体设计、生产其本身的智能卡产品时，将会把其本身的一些保密特性融入其产品中，例如加密的算法、认证的方式等。

读写机具向卡传送信号时，通过 13.65MHz 的射频载波信号，但采用的是异步、NRZ 编码方式，以及 10%ASK 传送的方案；在卡向读写机具传送信号时，则是采用的 BPSK 编码进行调制。信息比特 1 和信息比特 0 的区别在于信息 1 的信号幅度大，即信号强，信息 0 的信号幅度小，即信号弱。这种方式的优点是持续不断的信号传递，不会出现能量波动的情况。图 8-27 为从智能卡角度观察的 Type B 协议流程。

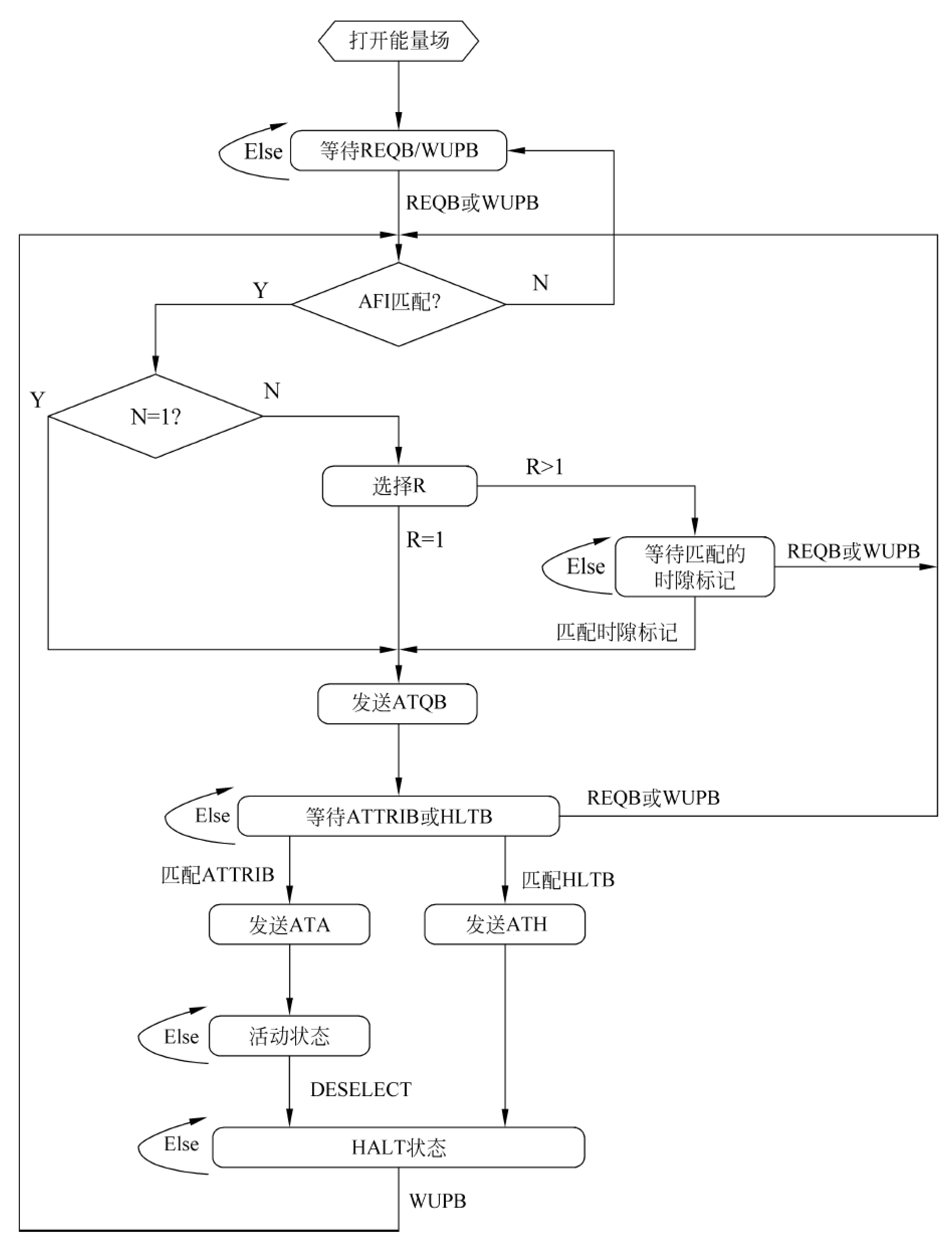


图 8-27 Type B 类型的智能卡状态转换流程

为了检测是否有智能卡进入读写器的有效区域，读写器要不停地通过天线发出查询命令，即REQB命令。当有卡进入有效区域时，该卡将对REQB命令作出应答ATQB，如果有两张或者更多的卡同时应答，便发生冲撞。由于每张卡回送的应答ATQB都包含卡片唯一的识别号(PUPI)，因此可以保证它们的应答数据ATQB不同，这样读写器便可顺利地检测到冲撞发生了。但由于冲撞的出现，读写器并不能从单张卡中获取有效的数据信息。只有通过Type B防冲撞机制，读写器才可以将这些卡分离开，与每张卡建立独立的通信通道，使得卡片之间互不干扰。

冲撞发生后，如果读写器要接收到卡片的应答数据，并与这些卡分别建立通信通道，必须让这些冲撞的卡在不同的时隙回送它们的应答数据ATQB，而不是同时回送。防冲撞方案以时隙为基础，时隙的个数由REQB命令中的参数$N(1\leqslant N\leqslant 16)$决定，所有的卡片都根据参数$N$生成随机数$R(R\leqslant N)$。接着读写器发送$N-1$个SLOTMARKER命令，用来规定每一个时隙，只有随机数R与SLOTMARKER命令定义的时隙序号相等的卡才回送ATQB应答。这样，读写器就有可能在同一时隙里只收到一张卡的应答，从而与它建立起独立的通信通道。如果凑巧出现几张卡产生的随机数R相等，它们在同一时隙内产生应答，便产生了第二次冲撞，在这种情况下，读写器可以重新定义一个N，再次发送REQB命令，重复上面所介绍的方法，直到与所有的卡都建立独立的通信通道为止。至此，防冲撞问题就解决了。

如果卡片与读写器之间建立了通信通道，我们称这张卡处于ACTIVE状态，它将不对REQB命令和SLOTMARKER命令产生任何反应。若读写器已经完成对某张卡的所有操作，可以对它发送HALT命令，使卡处于HALT状态。只有接收到WUPB命令，处于HALT状态的卡才可以被再次唤醒。在完成防冲撞过程之后，读写设备和智能卡之间就可以进行数据传输了。

Type B在机具和非接触式智能卡之间采用串行命令，如图8-28所示。

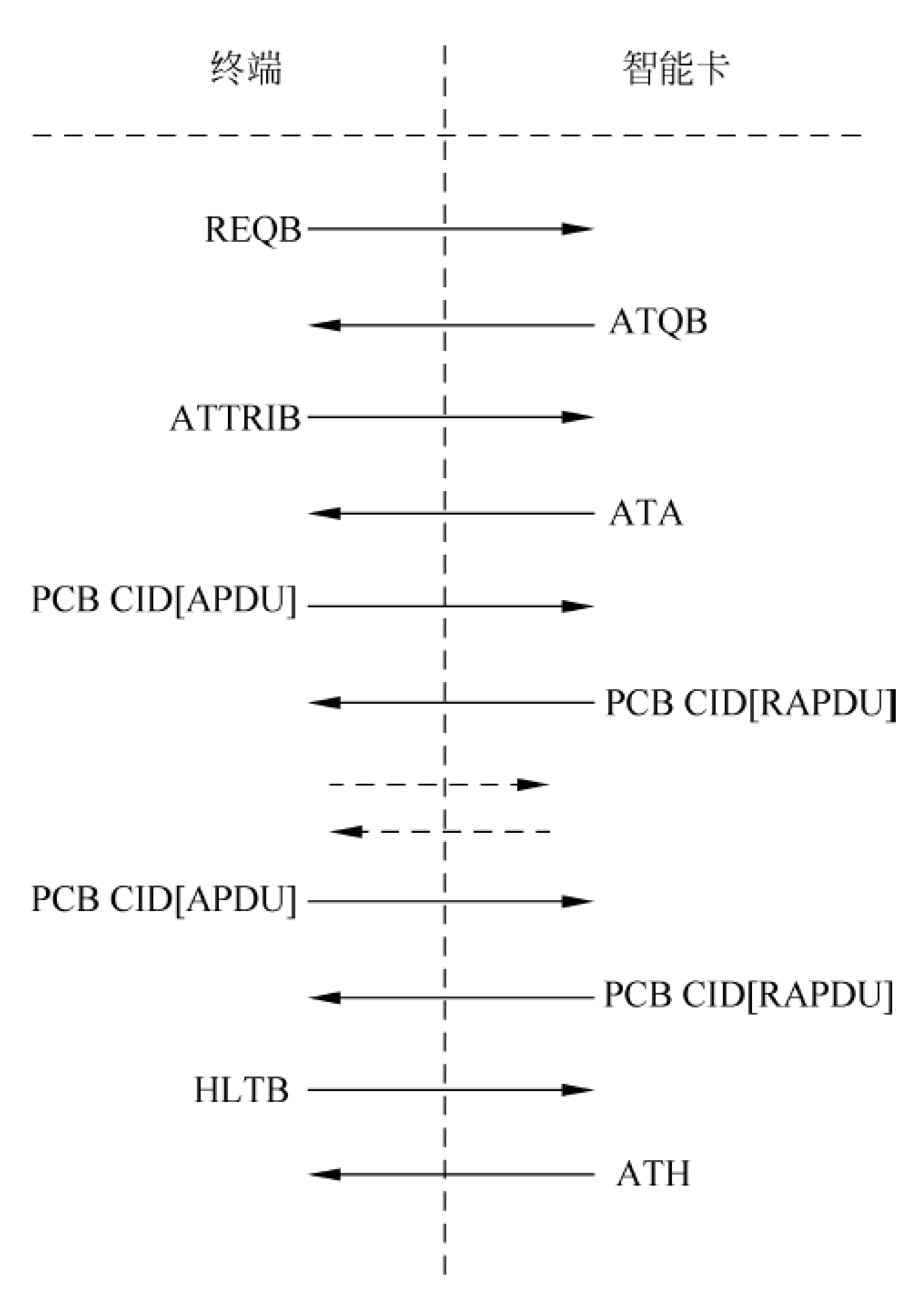


图8-28 Type B协议中正常流程的命令和响应

命令格式简单解释如下，更加详细的内容可参见ISO/IEC 14443-3/4规范。

(1) REQB/WUPB命令

PCD通过REQB命令请求建立通信，通过WUPB唤醒处于停止状态的PICC。REQB/WUPB命令格式参见表8-12。

表8-12 REQB/WUPB命令格式

1st字节	2nd字节	3rd字节	4th,5th字节
Apf 反碰撞前缀	AFI 应用系列标识符	PARAM 参数	CRCB 校验

续表

1^{st}字节	2^{nd}字节	3^{rd}字节				4th,5^{th}字节
‘05’	高四位＝应用类别编码 低四位＝具体应用编码	b8b7b6	b5	b4	b3 b2 b1	
		RFU	支持扩展 ATQB	REQB WUPB	时隙总数 N N＝1-16	

(2) ATQB 响应

PICC 通过 ATQB 响应应答 REQB 命令，ATQB 中含有 PICC 的应用和协议信息。ATQB 响应格式参见表 8-13。

表 8-13　ATQB 响应格式

1^{st}字节	2nd～5^{th}字节	6th～9^{th}字节	10th～12^{th}字节	13th～14^{th}字节
	PUPI	应用数据	协议信息	CRCB
反碰撞前缀‘50’	PICC 标识符唯一序列号或随机数	PICC 安装哪些应用	10th：位速率能力 b8＝双向是否一致 b7b6b5＝PICC 到 PCD 的速率 b4＝0 b3b2b1＝PICC 到 PCD 的速率 11th：b8b7b6b5＝最大帧大小 b4b3b2b1＝协议类型 12th：b8b7b6b5 ＝ 帧等待时间数(FWI) b4b3＝应用数据编码(ADC) b2b1＝帧选择(FO)	校验

(3) ATTRIB 命令

ATTRIB 命令是 PCD 用来进一步和 PICC 交互通信所需的信息，ATTRIB 命令格式参见表 8-14。

表 8-14　ATTRIB 命令格式

1st	2nd～5th	6th	7th	8th	9th	10th…	最后
‘1D’	PUPI	参数 1	参数 2	参数 3	参数 4	高层信息 ≥0 字节	CRCB 2 字节
	与 ATQB 中一致	传递 TR0、TR1、EOF、SOF	b4b3b2b1：FSDI ＝ PCD 可接受的最大帧长度 b8b7b6b5： 位传输速率选择	协议类型选择	b4b3b2b1：CID b8b7b6b5：0000		

(4) ATA 响应

PICC 接收到 PUPI 和校验码都正确有效的 ATTRIB 命令后，需回发 ATA 响应。ATA 响应的格式参见表 8-15 所示。

表 8-15 ATA 响应格式

1st字节		2nd…字节	最后 2 字节
MBLI	CID	高层应答	CRCB
最大缓存区长度系数	卡返回的卡标识符	与 ATTRIB 命令中的高层命令对应	校验

(5) HLTB 命令

PCD 发送 HLTB 命令可使得 PICC 进入停止状态。HLTB 命令的格式参见表 8-16 所示。

(6) ATH 响应

PICC 收到 PCD 发送的 HLTB 命令后,返回 ATH(answer to halt)响应,然后进入停止状态。此后 PICC 只能接收 WUPB 命令,忽略其他命令。ATH 响应的格式参见表 8-17 所示。

表 8-16 HLTB 命令格式

1st字节	2nd～5th字节	6th～7th字节
'50'	PUPI	CRCB

表 8-17 ATH 响应格式

1st字节	2nd～3rd字节
'00'	CRCB

(7) 数据交换命令和响应

激活卡片后,PCD 和 PICC 进行数据交换,命令与 Type A 协议一样,均采用数据帧格式,可参考本节的 Type A 协议部分内容。

8.3.3 整体结构

读写机具是应用软件与智能卡之间进行信息传输、运算和数据存储的中间媒介,起到桥梁作用。非接触式机具由接口控制器(RS232/USB 控制器)、微控制器、射频识别接口芯片和天线,共四部分组成。图 8-29 为通用非接触式读写机具的总体结构图。

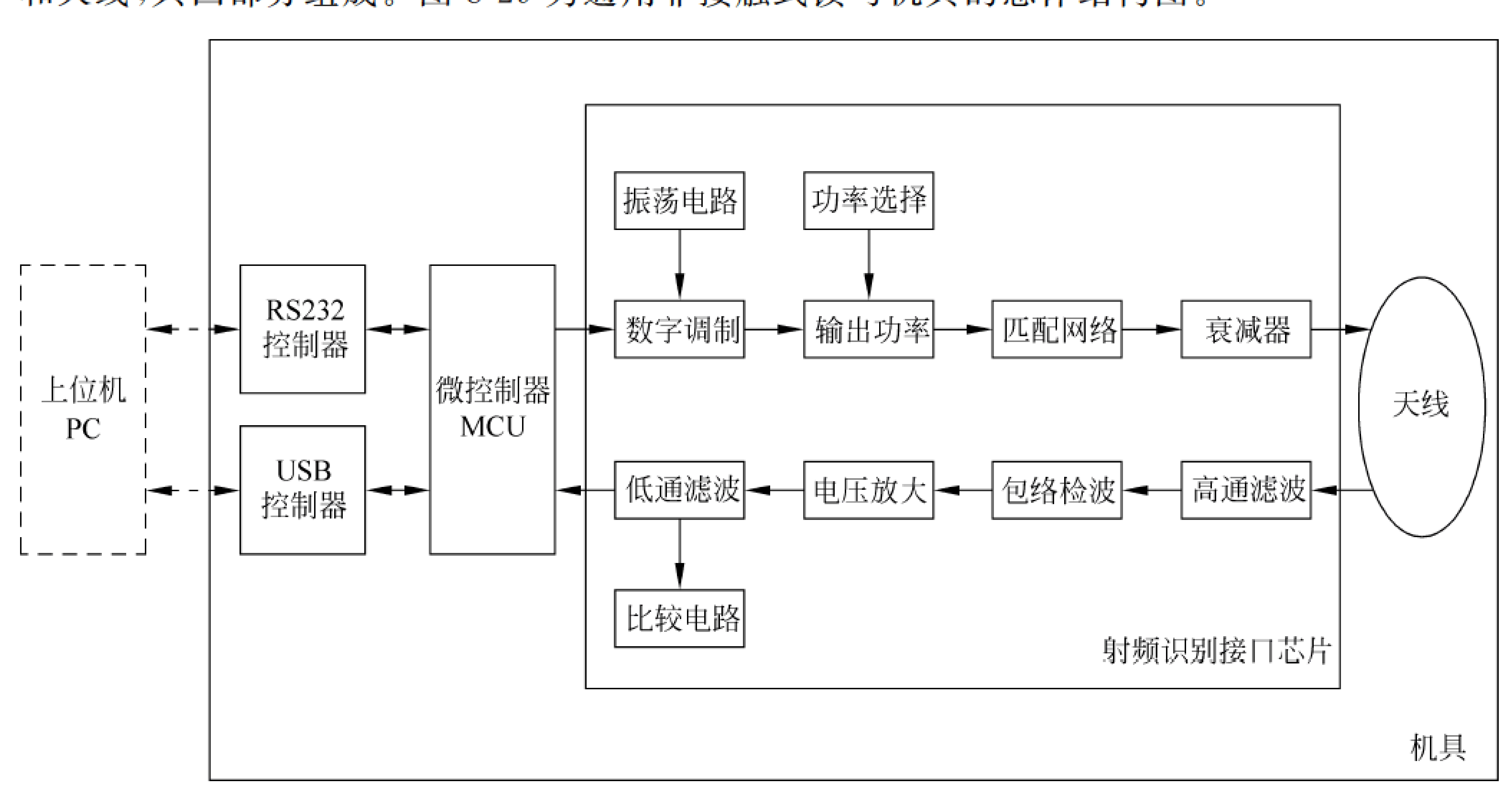


图 8-29 非接触式读写机具总体结构图

8.3.4 接口芯片

射频识别接口芯片的提供商非常多，比较典型的有NXP公司、上海华虹、ATMEL、uEM等公司，典型芯片有RC531、EM4094、SHC170X等系列芯片。

RC531是NXP公司开发的非接触式智能卡读卡器芯片系列中的一种，它可以读写符合ISO/IEC 14443标准的Type A和Type B卡，其内部自带的发射部件能够直接驱动天线，操作距离达到10cm，不需要增加额外的驱动电路。接收电路提供可靠的ISO/IEC 14443模拟信号的解调和解码，内部数字电路可以实现ISO/IEC 14443帧的处理和错误检测，灵活的并行接口可以和多种MCU相连，另外还支持SPI兼容接口。图8-30为RC531内部结构，欲了解更多RC531的内容可以参考NXP公司的产品手册。

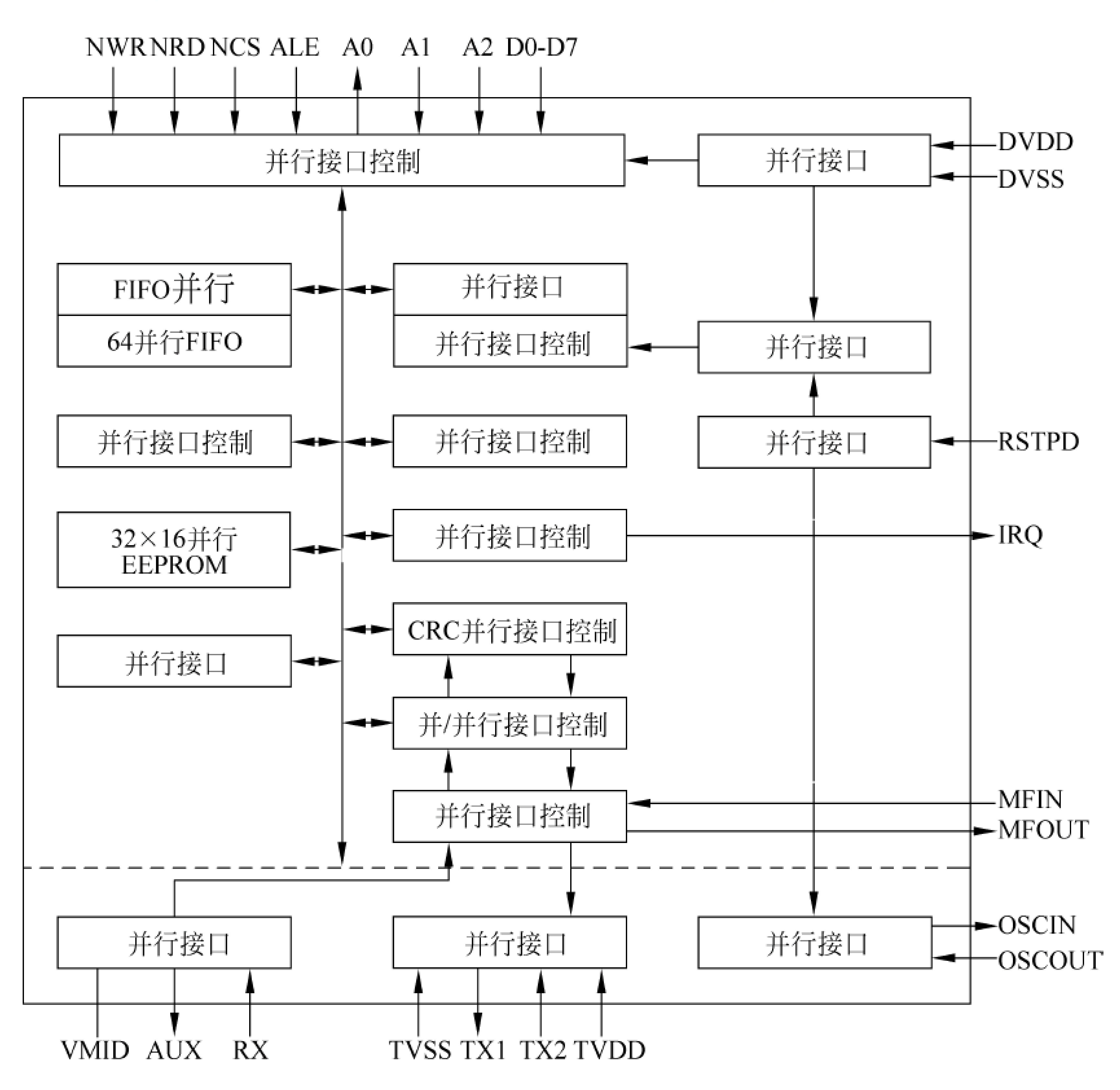


图8-30 RC531芯片结构框图

EM4094是瑞士uEM公司开发的一体化高集成读卡器芯片，适用于13.56MHz的RFID阅读器系统，该系统的副载波为212～848kHz，兼容ISO/IEC 14443和ISO/IEC 15693协议。其协议标准可通过一个3线的串口界面来编程31个配置位而完成。该芯片可用作低成本阅读器解决方案和手持式阅读器解决方案。更为详细的内容可参见EM4094芯片手册。

8.3.5 天线设计

智能卡读写设备天线的作用：

- 将高频接口输出的高频电流转换为电磁波，并向空间发送，传输方向为读写设备到卡片。
- 将从空间接收到的电磁波转换成可供高频接口芯片识别处理的高频电流，传输方向是卡片到读写设备。

非接触式智能卡与读写设备之间存在两种耦合方式：

(1) 电感耦合(变压器模式)。适用于近场识别(卡机之间的通信距离远小于载波的一个波长)。如ISO/IEC 14443规范中的载波频率为13.56MHz，其波长为22.1m。读写设备将能量和信息，通过天线上的电压(或电流)的变化形成的交变磁场传送到近耦合卡片的天线，并以感应电压(或电流)的变化予以表现。而近耦合卡片的数据发送则通过负载调制，并经交变磁场将此调制编号反映到读写设备天线，变现为反作用的感应电压(或电流)变化。

(2) 电磁反向散射耦合(雷达模式)。电磁反向散射耦合利用的是读写设备天线辐射的交变电磁能，读写设备天线发射的电磁波照射到远耦合卡片，形成反射回波，并被读写设备天线接收。读写设备的发射信息是通过读写设备天线发射电磁波的幅度、频率和相位的调制予以传送的。

天线对于读写器和非接触式智能卡卡体都非常重要，天线设计是否合理对系统性能影响巨大。电感耦合和电磁反向散射耦合两种方式的天线设计也不尽相同。下面以电感耦合设计为例稍加介绍。

从电磁感应原理和通信原理分析，天线设计的原则：

(1) 天线电流尽可能大，以便尽可能地产生大的磁通量，用于能量和信息的传递。

(2) 带宽足够宽，以便信息的无失真传递。

天线设计过程中的两个关键参数：

(1) 谐振频率 f_0 表征的物理量

① 卡耦合到的能量

② 卡反射的能量

③ 卡返回信号的传输特性(波形)

(2) 品质因数 Q 表征的物理量

① 卡耦合到的能量

② 信噪比

③ 卡返回信号的传输特性(波形)

符合ISO/IEC 14443规范的非接触式智能卡等效电路如图8-31所示。

其中谐振频率 f_0 计算公式

$$f_0 = \frac{1}{2\pi \times \sqrt{L_{线圈} \times (C_{线圈} + C_{封装} + C_{模块} + C_{IC})}}$$

读写器等效电路如图8-32所示。

一般情况下，回路中总电阻的大小远小于回路中的总阻抗，所以谐振频率 f_0 取决于电

感和电容，计算公式为

$$f_0 = \frac{1}{2\pi \times \sqrt{L_{线圈} \times C_{线圈}}}$$

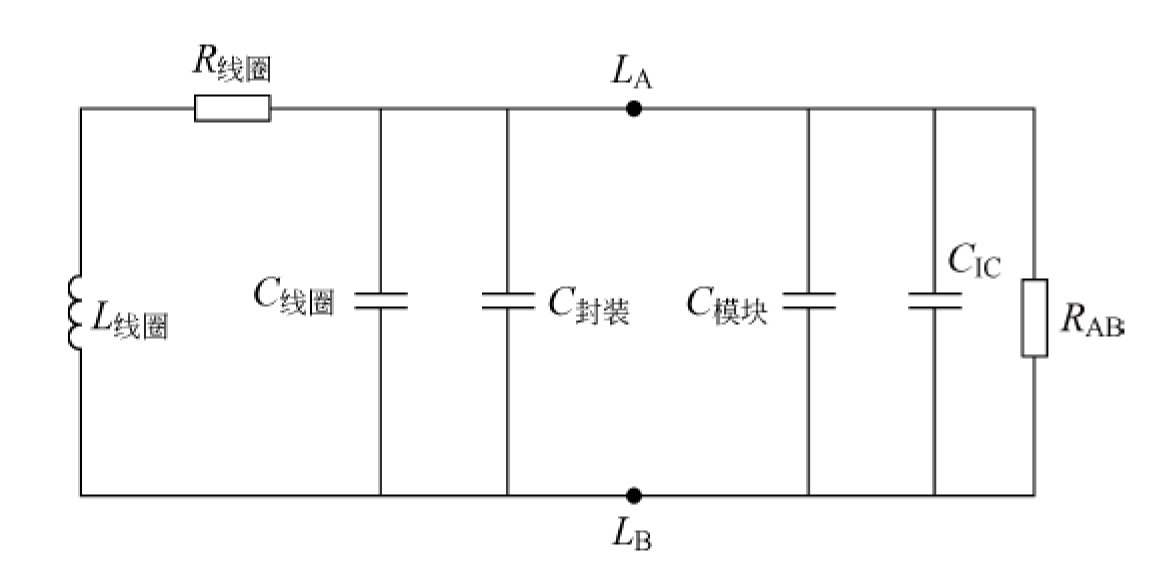


图 8-31 非接触式智能卡等效电路一

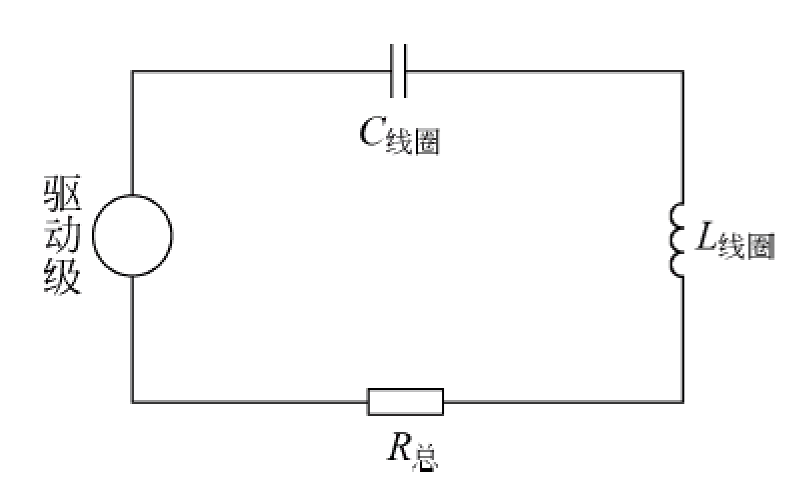


图 8-32 非接触式智能卡等效电路二

品质因数 Q 是表征一个储能器件（如电感线圈、电容等）、谐振电路所储能量同每周损耗能量之比的一种质量指标。元件的 Q 值愈大，用该元件组成的电路或网络的选择性愈佳。电抗元件的 Q 值等于它的电抗同等效串联电阻的比值。谐振回路的品质因数为谐振回路的特性阻抗与回路电阻之比。即

$$Q = \frac{2\pi f_0 L}{R} = \frac{1}{2\pi f_0 CR} = \frac{\sqrt{L/C}}{R}$$

设计过程中，Q 值越大，天线电压也越高，谐振电流越大，有利于能量传输。但是天线的传输带宽 $W=\frac{f_0}{Q}$，Q 值的增大可使得带宽减小，系统更容易受频率失配的影响，明显减弱卡片接收到的调制边带。因此 Q 的取值范围设计经验值通常为 10～30。在设计过程中，上述的理论计算只能辅助设计，并且要牢记：

- 电感值随频率变化；
- 电容随电压值变化；
- 谐振频率只在某个特定的场强下有效；
- 负载调制幅度随场强变化。

8.4 应用接口设计

智能卡应用系统是紧密贴近用户业务需求的软件实现，主要功能不外乎提供人机操作界面、数据库和网络等功能，能够支持一种或多种读写机具，运行在 Windows 系统或者其他操作系统中。

应用层代码一般使用一套操作系统支持的 API 函数。设备驱动通过在应用层和机具硬件专用代码之间的转化来完成它的任务。机具硬件代码处理那些访问外设电路的必要协议，包括监测状态信号和在合适的时间切换控制信号。

从通信的硬件角度来看，一些设备驱动是处理所有事情的单片机电路驱动，它们处理的事情包括从与应用程序的通信到对连接到硬件的端口或内存地址的读写操作。

下面主要介绍应用程序中对机具的两种访问方式，而人机操作界面、数据库和网络等上

层功能开发的书籍和资源非常丰富，本书不再赘述。

8.4.1 PC/SC 规范

PC/SC 规范由微软公司与世界其他著名的智能卡厂商组成的 PC/SC 工作组提出的。PC/SC 规范是一个基于 Windows 平台的标准用户接口(API)，提供了一个从个人计算机(personal computer)到智能卡(smart card)的整合环境，使智能卡进入 PC 的问题变得更容易解决。PC/SC 的主要优点就是让应用程序不必为了与智能卡通信而去了解智能卡读卡器的细节，并且应用程序还能适用于任何遵从 PC/SC 标准的读卡器。

虽然到目前为止，Windows 是唯一支持 PC/SC 标准的操作系统平台，但由于 Windows 的影响力，PC/SC 规范也为智能卡业界所接收。到目前为止，PC/SC 规范的最新版本是 PC/SC 2.0 规范，更多细节可参见第 12 章。

PC/SC 规范建立在工业标准——ISO/IEC 7816 和 EMV 标准的基础上，但它对底层的设备接口和独立于设备的应用 API 接口(例如用来允许多个应用共享使用系统同一张智能卡的资源管理器)做了更详尽的补充。它的提出主要是为了达到以下目标：

(1) 遵从现在智能卡和 PC 的标准并在适当的地方给予扩充。

(2) 跨平台的可操作性，该规范可在多种硬件和软件平台上实现，这使得应用程序可以采用不同厂商提供的产品。

(3) 不需要写应用代码就可以享受技术进步的好处。

(4) 建立应用级的智能卡服务接口，推广 ICC 在 PC 上的应用，并促成 PC 采用 ICC 做主标准设备。

PC/SC 体系参见图 8-33。PC/SC 体系由硬件和软件两部分组成，其中软件部分主要由三个主要部件组成，分别描述如下：

(1) 智能卡读写器驱动(smart card reader driver)是由读写器厂商提供的可安装部件。

(2) 智能卡资源管理器(smart card resource manager)使用 Win32 API 函数实现，是由操作系统厂商提供的系统级部件。智能卡资源管理器是 PC/SC 体系结构的核心部分。它在结构中处于应用程序和智能卡之间的位置，允许并发访问一个给定的读卡器。

(3) 智能卡服务提供者(smart card service providers)，服务程序是由厂商提供的可安装部件，用于提供访问特殊服务的手段，其使用的是基本 COM 的界面方式。

PC/SC 的 API 函数由操作系统提供，函数声明在 Winscard.h 中，所用的库是 Scarddlg.lib。由生产厂商提供的智能卡读写器驱动程序负责把资源管理器语言转化为读卡器语言。

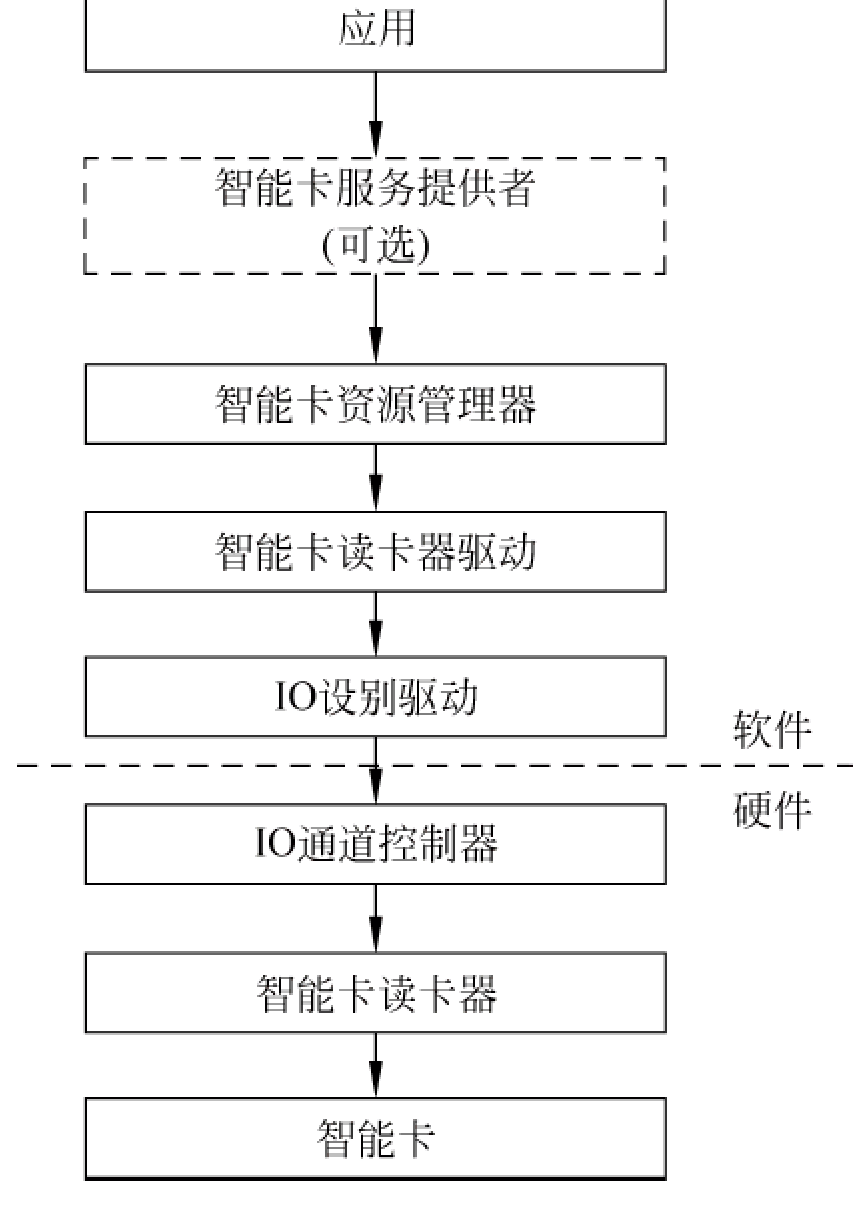


图 8-33 PC/SC 规范框架图

8.4.2 PC/SC 接口开发范例

利用 PC/SC 的 API 和符合 PC/SC 协议的机具进行通信非常方便，主要有 4 个关键步骤：

(1) 遍历当前所有读卡器，并试图建立连接。

```
SCARDCONTEXT  m_hContext;                                    //卡设备上下文
res   = ::SCardListReaders(m_hContext,sReader,(LPTSTR)&pmszReaders,&cch);
```

(2) 建立连接。

```
SCARDHANDLE     m_hCard;                                     //读卡器句柄
::SCardConnect(m_hContext,sReaderName,SCARD_SHARE_SHARED,SCARD_PROTOCOL_T0|SCARD_PROTOCOL_
T1|SCARD_PROTOCOL_RAW,&m_hCard,&dwAP);
```

(3) 应用发送给机具的 APDU 命令存储在 ucCmd 缓冲区中，发送长度为 uCmdLen，接收数据存储在 ucRes 中。

```
      SCARD_IO_REQUEST pciOut;
//读卡器的传输协议为 T1 协议
      memcpy(&pciOut,SCARD_PCI_T1,sizeof(SCARD_IO_REQUEST));
//例如随机数命令：
      memcpy( ucCmd , "\x00\x84\x00\x00\x08", 5);
      nCmdLen = 5;
//参数为：读卡器句柄，协议类型，发送缓冲区，发送长度
//请求结构，接收缓冲区，接收长度
      res = ::SCardTransmit(m_hCard,SCARD_PCI_T1,ucCmd,nCmdLen, &pciOut ,ucRes,&nResLen);
```

(4) 使用完设备后，需断开连接。

```
::SCardDisconnect(m_hCard,SCARD_LEAVE_CARD);
```

下面是实际工程开发过程中形成的 PC/SC 功能的 C++类的实现代码。

```
////////////////////////////////////////////////////////////////////////////
#ifndef __CPCSC_HEADER__
#define __CPCSC_HEADER__

#include <winscard.h>
#pragma comment(lib, "winscard")

class CPCSC
{
public:
     CPCSC();
     virtual ~CPCSC();

     bool  ConnectCard(char *   sReaderName   =   NULL, char *   sReader   =   NULL);
     void  DisConnectCard();

public:
```

```
    bool SendRevAPDU(IN unsigned char * ucCmd, IN DWORD nCmdLen,
                     OUT unsigned char * ucRes, IN OUT DWORD& nResLen);

    void GetPCSCInfo( CString &strATR, BOOL& isT0);

protected:
    SCARDCONTEXT  m_hContext;                               //卡设备上下文
    SCARDHANDLE    m_hCard;                                 //读卡器句柄
    CString m_cardName;                                     //读卡器名称
    BOOL m_isT0;

};
#endif //__CPCSC_HEADER__

///////////////////////////////////////////////////////////////////////////
#include "StdAfx.h"
#include "PCSC.h"

CPCSC::CPCSC()
{
    m_hContext   =   0;
    m_hCard      =   0;
    m_cardName = "";
}

CPCSC::~CPCSC()
{
    DisConnectCard();
}

//连接设备
bool CPCSC::ConnectCard(char * sReaderName, char *  sReader)
{
    DisConnectCard();

    DWORD dwAP;
    LONG  res;

    //建立设备上下文
    res  =  ::SCardEstablishContext(SCARD_SCOPE_USER,NULL,NULL,&m_hContext);
    if(res  !=  SCARD_S_SUCCESS)
    {
        m_hContext  =  0;
        return  false;
    }

    LPTSTR  pmszReaders  =  NULL;
    LPTSTR pReader       =  NULL;
    DWORD  cch           =  SCARD_AUTOALLOCATE;
    bool  bConnected  =  false;
```

```
if( sReaderName == NULL )
{
    //遍历当前所有读卡器,并试图建立连接
    res  =  ::SCardListReaders(m_hContext,sReader,(LPTSTR)&pmszReaders,&cch);

    if(res ==   SCARD_S_SUCCESS)
    {
        pReader  =  pmszReaders;

        while ( '\0'!=  * pReader )
        {
            //得到一个读卡器名字的串
            m_cardName = pReader;
            res  =  ::SCardConnect(m_hContext,pReader,SCARD_SHARE_SHARED,
                       SCARD_PROTOCOL_T0|SCARD_PROTOCOL_T1|SCARD_PROTOCOL_RAW,
                       &m_hCard,&dwAP);

            if(SCARD_S_SUCCESS  ==  res)
            {
                bConnected  =  true;
                break;
            }
            else
                pReader = pReader + strlen(pReader) + 1;
        }

        //释放名字串空间
        ::SCardFreeMemory(m_hContext,pmszReaders);
    }
}
else
{
    m_cardName = sReaderName;
    res = ::SCardConnect(m_hContext,sReaderName,
                       SCARD_SHARE_SHARED,
                       SCARD_PROTOCOL_T0|SCARD_PROTOCOL_T1|SCARD_PROTOCOL_RAW,
                       &m_hCard,&dwAP);

    if(SCARD_S_SUCCESS  ==  res)
        bConnected  =  true;
    else
        bConnected  =  false;
}

if(!bConnected)
{
    //连接读卡器失败,释放设备上下文
    ::SCardReleaseContext(m_hContext);
    m_hContext  =  0;
    m_hCard    =  0;
}
```

```
    if( dwAP == SCARD_PROTOCOL_T0)
        m_isT0 = TRUE;
    else if( dwAP == SCARD_PROTOCOL_T1)
        m_isT0 = FALSE;

    return bConnected;
}

//断开连接
void CPCSC::DisConnectCard()
{
    if(m_hCard)
    {
        ::SCardDisconnect(m_hCard,SCARD_UNPOWER_CARD);
        m_hCard     =   0;
    }
    if(m_hContext)
    {
        ::SCardReleaseContext(m_hContext);
        m_hContext   =   0;
    }
}
//获取卡片信息
void CPCSC::GetPCSCInfo( CString &strATR, BOOL& isT0)
{
    isT0 = m_isT0;
    SCARD_READERSTATE Reader[1];

    Reader[0].dwCurrentState = SCARD_STATE_UNAWARE;
    Reader[0].dwEventState = SCARD_STATE_UNAWARE;
    Reader[0].szReader = m_cardName;
    long res = SCardGetStatusChange( m_hContext, 300, Reader, 1);

    if( res == SCARD_S_SUCCESS )
    {
         strATR = "复位信息: ";
         CString strTemp;
         for ( int i = 0; i < Reader -> cbAtr; i++)
         {
             strTemp.Format(" %02X ",Reader -> rgbAtr[i]);
             strATR += strTemp;
         }
    }
    else
    {
        strATR += "复位信息: Error!";
    }
}
////////////////////////////////////////////////////////////////////////////
```

8.4.3 非 PC/SC 接口开发范例

除符合 PC/SC 规范的机具外，还有一类非 PC/SC 规范的机具，如握奇数据有限公司的 CRW-X 机具。

CRW-X 机具为非接触式通用读卡器，符合中国银联规范，并支持 ISO/IEC 14443 Type A/Type B 的非接触 CPU 卡和 Mifare one 卡，内置两个符合 ISO/IEC 7816-3 的 SAM 小 IC 卡，支持多种通信接口和通信协议，功能齐全，外观漂亮，性能稳定，质量可靠，适用于各种非接触式 IC 卡的应用系统。

1. 主要功能

(1) 支持符合 ISO/IEC 14443 Type A/Type B 的非接触式卡。

(2) 支持 Mifare one 卡。

(3) 内置两个小 SAM 卡座，支持符合 ISO/IEC 7816 的 CPU 卡。

(4) 支持串口通信，支持二进制和 ASCII 码两种通信方式。

(5) 支持全速 USB 通信，速率 12Mbps。

(6) 内置蜂鸣器，用户可以控制。

(7) 3 个指示灯(蓝/红/绿)，用户可控制两个(红/绿)。

(8) 一个可选的 ESAM 芯片，通信符合 ISO/IEC 7816 标准。

(9) 一个可选的扩展的门禁接口。

(10) 一个可选的外接扩展的大天线。

(11) 读卡器的固件程序可通过用户升级来增加新功能。

2. 符合标准

(1) 非接触 IC 卡读卡器技术规范。

(2) ISO/IEC 14443-1/2/3/4。

(3) ISO/IEC 7816-1/2/3。

(4) USB2.0 标准。

(5) EIA-232-E 串口通信标准。

CRW-X 机具有详细的用户手册和用户函数库，手册中给出了全面的 API 函数说明。

下面是实际工程开发过程中形成的使用 USB 接口的 CRW-Xu 机具功能的 C++类的完整实现代码。

```
////////////////////////////////////////////////////////////////////////////////
#ifndef __CCRWXREADER__HEADER__
#define __CCRWXREADER__HEADER__
#include <string>
#include <map>
using namespace std;

#include "WDCRW.h"
#include "assistant.h"

#define COMMAND_EXECUTE_ERROR         -1
#define COMMAND_NOT_EXIST           -2
```

```
#define COMMAND_FORMAT_ERROR        -3

class CCRWXReader
{
public:
      CCRWXReader();

      //打开机具
      bool Open();
      bool Close();

      //自动复位,包括 REQ/防碰撞等流程
      bool AUTORESET();

      //传输数据,参数为:发送数据缓冲区,发送数据长度,接收数据缓冲区,接收长度
      bool TransmitData( unsigned char * out, unsigned int outlen, unsigned char * in, unsigned
char *  inlen);
private:
      HANDLE m_hKey;
      CWDCRW m_wdcrw;
};
#endif
////////////////////////////////////////////////////////////////////////////
#include "stdafx.h"
#include "CCRWXReader.h"

CCRWXReader::CCRWXReader()
{
   m_hKey = INVALID_HANDLE_VALUE;
}

//打开机具
bool CCRWXReader::Open()
{
      //默认打开 USB1
      m_hKey = m_wdcrw.CT_open("USB1",0,0);
      if( m_hKey == INVALID_HANDLE_VALUE)
      {
            m_hKey = m_wdcrw.CT_open("USB2",0,0);

            if(m_hKey == INVALID_HANDLE_VALUE)
            {
                  m_hKey = m_wdcrw.CT_open("USB3",0,0);
            }

            if(m_hKey == INVALID_HANDLE_VALUE)
            {
                  return false;
            }
      }
      else
```

```
    {
        m_wdcrw.ICC_set_NAD(m_hKey,0x15);
        return true;
    }

    return false;
}
bool CCRWXReader::Close()
{
    return true;
}

//自动复位,包括 REQ/防碰撞等流程
bool CCRWXReader::AUTORESET()
{
    m_sLen = 5;
    memcpy(m_sBuff,"\x00\x12\x00\x00\x00",5);

    return TransmitData( m_sBuff,m_sLen,m_rBuff,&m_rLen);

}
//传输数据,参数为:发送数据缓冲区,发送数据长度,接收数据缓冲区,接收长度
bool CCRWXReader::TransmitData( unsigned char * out,unsigned int outlen,unsigned char * in,
unsigned char * inlen)
{
    long lRet = 0;
    lRet = m_wdcrw.ICC_tsi_api ( m_hKey,outlen,out,inlen,in);

    in[ * inlen] = (unsigned char)((lRet&0xFF00)>> 8);
    in[ * inlen + 1] = (unsigned char)(lRet&0x00FF);
    * inlen = * inlen + 2;

    return true;
}
/////////////////////////////////////////////////////////////////////////////
```

参考文献

[1] International Standard ISO/IEC 7816-4. Identification Cards. Integrated Circuit cards, Part 4: Organization, Security and Commands for Interchange, 2005

[2] EMV2000 Integrated Circuit Card Specification for Payment Systems. Book 2-Security and Key Management, 2004

[3] ICAO Doc 9303. Machine Readable Travel Documents, Fifth Edition, 2003

[4] Technical Report: PKI Digital Signatures for Machine Readable Travel Documents, version 4, 2003

[5] ICAO Doc 9303. Part 1: Machine Readable Passports, Sixth Edition, 2006

[6] Technical Report: Development of a Logical Data Structure, LDS for Optional Capacity Expansion Technologies, Vefision 1.7, 2004

[7] International Standard ISO/IEC 14443-3. Identification Cards. Contactless Integrated Circuit (s)

Cards, Proximity Cards, Part 3: Initialization and Anti-collision, 2001
[8] International Standard ISO/IEC 14443-4. Identification Cards. Contactless Integrated Circuit (s) Cards, Proximity Cards, Part 4: Transmission Protocol, 2001
[9] International Standard ISO/IEC 10373-6. Identification cards. Test Methods, Part 6: Proximity Cards, Amendment 1: Protocol Test Methods for Proximity Cards, 2001
[10] 胥怡心,张其善. 智能 IC 卡文件系统的设计与实现. 微计算机应用,2007,28(1):83～86
[11] 黎妹红,张其善. 智能卡双界面操作系统的设计与实现. 计算机工程与应用,2003,39(27):138～140
[12] 倪志鸿. 智能卡技术的研究与开发. 北京:北京航空航天大学,2002
[13] 王爱英. 智能卡技术. 第 2 版. 北京:清华大学出版社,2000
[14] 陆永宁. 非接触式 IC 卡原理与应用. 北京:电子工业出版社,2006
[15] 李胜广,张其善,张江波. 手持式高速 IC 卡读卡器设计. 电子测量技术,2006,29(3):55～56
[16] 严光文,张其善. 射频识别卡读写模块的设计. 北京航空航天大学学报,2003,29(2):178～180
[17] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 2: Interface Requirements for Compatible IC Cards and Readers, 2005
[18] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 3: Requirements for PC-Connected Interface, 2007
[19] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 4: IFD Design Considerations and Reference, 2005
[20] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 5: ICC Resource Manager Definition, 2005
[21] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 6: ICC Service Provider Interface Definition, 2005
[22] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 7: Application Domain and Developer Design, 2005
[23] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 8: Recommendations for ICC Security and Privacy Devices, 2005
[24] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Pan 9: IFDs with Extended Capabilities, 2005
[25] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems, Part 10: IFDs with Secure Pin Entry Capabilities, 2005



CSP 应用与开发

加密技术发展到今天，涵盖的内容十分广泛，产生了大量的加密算法、协议和标准，这对应用加密技术提出了严峻的挑战。如何将加密技术的应用从复杂的实现方法中分离出来是个亟待解决的问题。加密服务提供者(cryptographic service provider，CSP)则应运而生，它为加密应用提供服务，将加密应用与加密的实现分离开来，简化了对加密技术的应用。应用 CSP 技术，应用程序开发人员不需要对加密算法或协议有深入的理解，只需要了解 CSP 提供的服务功能和接口，就可以快速地开发出各种应用程序。

CSP 从功能上而言，提供了一套加密服务；从实现上而言，通常采用软件或硬件(智能卡、USBKey 等)的形式。大部分 CSP 都是针对特定的应用环境开发，并且至少包含一个动态链接库文件(dynamic-link library，DLL)。动态链接库用于实现 CSP 提供的加密服务，或者作为操作系统提供接口与真正提供加密服务的 CSP 进行通信。

CSP 离我们的日常工作和生活并不遥远，最常用的 Windows 操作系统就提供了许多可供用户使用的 CSP，通过在注册表编辑器中查看 HKLM\Software\Microsoft\Cryptography\Defaults\Provider 子键，可以看到微软提供的 CSP，如图 9-1 所示。其中包括有微软提供的 CSP，如 Microsoft Base Cryptographic Provider、Microsoft Enhanced Cryptographic Provider、Microsoft DSS Cryptographic Provider 等，也有其他公司提供的 CSP，如 Infineon SICRYPT Base Smart Card CSP 等，默认 CSP 是微软提供的 Microsoft Base Cryptographic Provider v1.0。

本章首先从 CSP 的系统架构入手，对 CSP 进行深入剖析；然后介绍如何应用加密应用

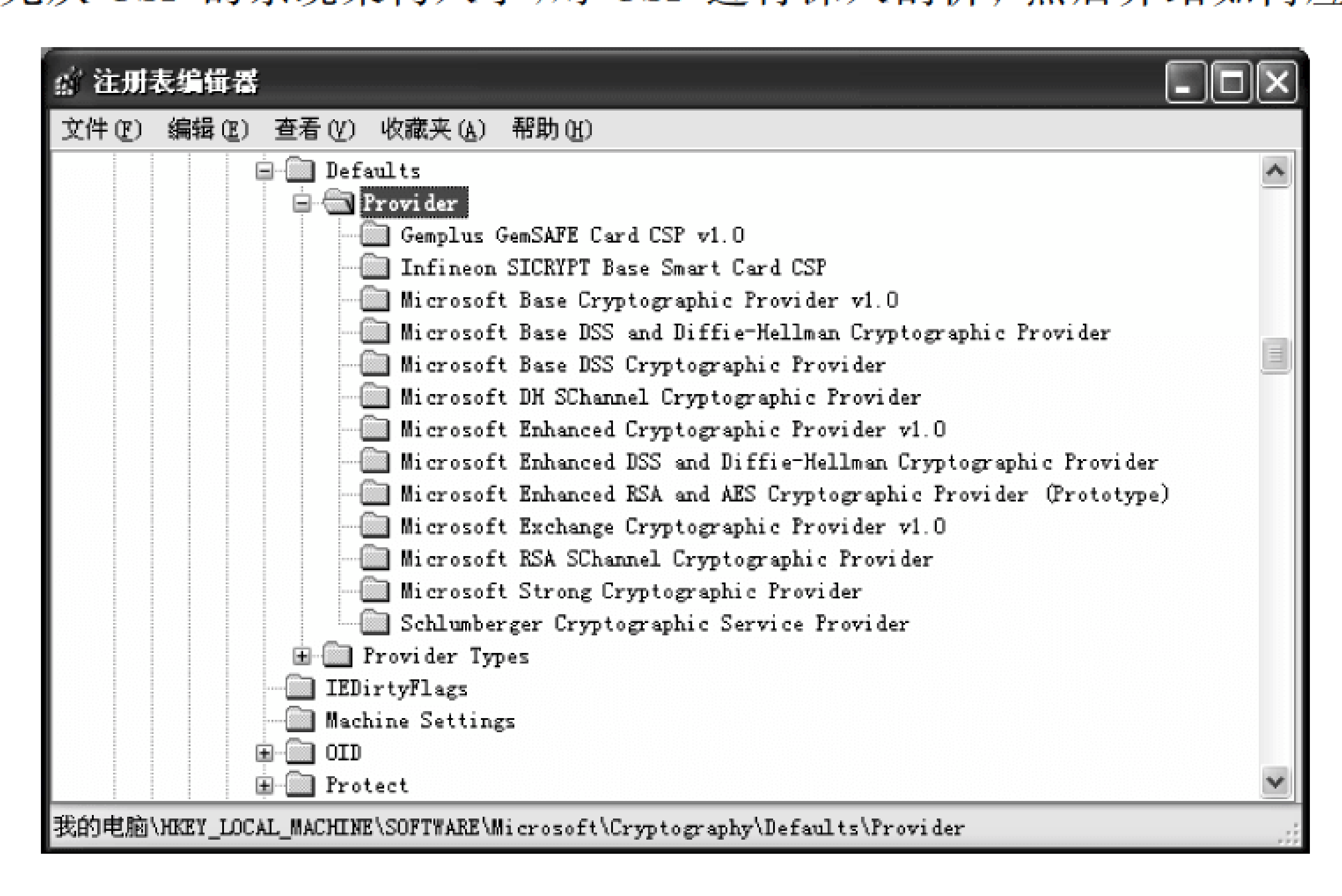


图 9-1　微软提供的 CSP

程序接口(cryptographic application programming interface,CryptoAPI)开发应用程序,对常见的CryptoAPI函数进行了简要说明,并通过两个开发实例介绍了如何应用CryptoAPI;最后介绍了开发CSP的基本流程,并对加密服务程序接口(cryptographic service programming interface,CryptoSPI)函数进行简要说明。

9.1 CSP介绍

9.1.1 CSP系统的架构

一个典型的CSP系统,分为应用层、操作系统层和加密服务提供层三部分,如图9-2所示。

应用层是针对具体应用开发的加密应用程序,通过操作系统提供的接口函数与操作系统进行通信。应用程序的开发者无需了解加密的具体实现过程。

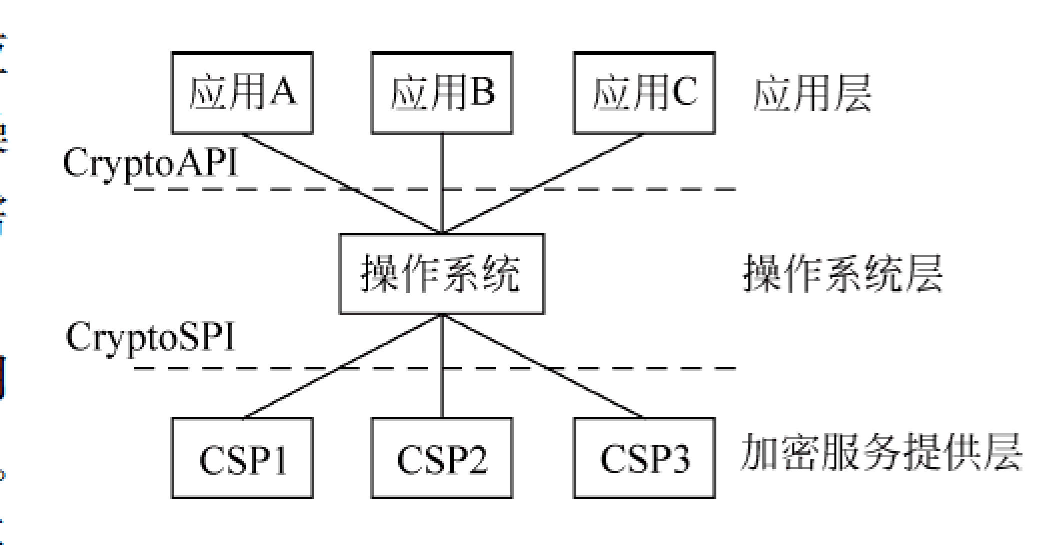


图9-2 典型CSP系统架构图

CryptoAPI是操作系统提供给上层应用程序的接口,它由一组API函数组成。CryptoAPI把应用程序的调用转交给操作系统,完成加密、认证、编码、证书的导入导出等加密服务。应用程序开发人员需要依照CryptoAPI,才能开发出正确的应用程序。CryptoAPI将CSP与应用程序隔离开来,即使CSP由于需求变化需要改动,但由于CryptoAPI接口没有变化,应用程序也无需改动。

操作系统层介于应用层和加密服务提供层之间,它将应用程序对CryptoAPI的调用转换为CryptoSPI的调用,驱动CSP执行实际操作,并将CSP执行后返回的结果经CryptoSPI由CryptoAPI返回给应用程序。同时,操作系统层还负责管理CSP的功能。

CryptoSPI是一组由操作系统提供给加密服务开发商的系统接口,它由一组API函数组成,加密服务开发商必须按照CryptoSPI的接口标准来开发CSP的各功能模块,操作系统通过CryptoSPI来实现对CSP的各种操作。

加密服务提供层提供一套基于软件或硬件的加解密服务。它可能是一套软件程序,也可能是一个硬件如智能卡、USBKey、加密板卡或加密机,还有可能是一套软件和硬件结合的系统,它的作用就是把真正的加解密操作封装起来,对外只开放接口。从外部来看,CSP就是一个黑盒,CSP的应用者只需了解它所提供的功能,了解各接口的输入和输出,即可方便快捷地使用CSP所提供的服务。CSP的出现简化了应用各种复杂加密技术的成本,并且由于加密操作过程对外部是不可见的,提高了加解密服务的安全性。

9.1.2 CSP的分类

每个CSP都有唯一的名称和类型。名称用于标识CSP,CSP类型用于说明该CSP支持的算法和提供的服务。依据所支持的密钥交换、数字签名、加密运算和Hash运算的算法不同,目前已经定义了10种CSP类型,且数量还在不断地增加,其支持的算法如表9-1所示。各种类型CSP的功能和特点如下。

(1) PROV_RSA_FULL 支持数据签名和加密。它是一个通用的 CSP 类型,所用的公钥操作均支持 RSA 公钥算法。

(2) PROV_RSA_AES 支持数据签名和加密。它是一个通用的 CSP 类型,所用的公钥操作均支持 RSA 公钥算法。它与 PROV_RSA_FULL 相比,增加了加密运算时对 AES 算法的支持。

(3) PROV_RSA_SIG 是 PROV_RSA_FULL 的子类型,支持数字签名和 Hash 运算。

(4) PROV_RSA_SCHANNEL 支持 RSA 和 Schannel 协议。

(5) PROV_DH_SHANNEL 支持 Diffie-Hellmanhe 和 Schannel 协议。

(6) PROV_FORTEZZA 支持一组由 NIST 拥有的加密算法和协议。

(7) PROV_MS_EXCHANGE 是根据微软电子邮件交换和与微软电子邮件兼容的其他应用的加密服务而设计的。

(8) PROV_ SSL 支持安全套接字(security socket layer,SSL)协议。

(9) PROV_DSS 同 PROV_RSA_SIG 类似,只支持数字签名和 Hash 运算,它支持的签名算法为 DSA(digital signature algorithm)。

(10) PROV_DSS_DH 是 PROV_DSS 的一个超集。

表 9-1　各种类型的 CSP

CSP 类型	交换算法	签名算法	对称加密算法	Hash 算法
PROV_RSA_FULL	RSA	RSA	RC2 RC4	MD5 SHA
PROV_RSA_AES	RSA	RSA	RC2 RC4 AES	MD5 SHA
PROV_RSA_SIG	none	RSA	none	MD5 SHA
PROV_RSA_SCHANNEL	RSA	RSA	RC4 DES Triple DES	MD5 SHA
PROV_DSS	DSS	none	DSS	MD5 SHA
PROV_DSS_DH	DH	DSS	CYLINK_MEK	MD5 SHA
PROV_DH_SCHANNEL	DH	DSS	DES Triple DES	MD5 SHA
PROV_FORTEZZA	KEA	DSS	Skipjack	SHA
PROV_MS_EXCHANGE	RSA	RSA	CAST	MD5
PROV_SSL	RSA	RSA	Varies	Varies

9.1.3 CSP 密钥库的逻辑结构

每个 CSP 都包括有一个密钥库(key database),用于存储使用该 CSP 的所有用户的密钥。通常,密钥库在安装 CSP 的时候就已经创建。每个密钥库由一个或多个密钥容器(key container)组成,密钥容器是密钥库中的逻辑存储器,用于保存某个特定用户的密钥。在应

用 CSP 时,需要针对不同的用户创建密钥容器,用来保存该用户使用 CSP 服务过程中的密钥。只有在销毁密钥容器时,保存的密钥才会销毁。图 9-3 说明了密钥库的构成,密钥库由一个或多个密钥容器组成,容器中保存了该 CSP 所支持的密钥对,如用于计算签名和密钥交换时的密钥对。

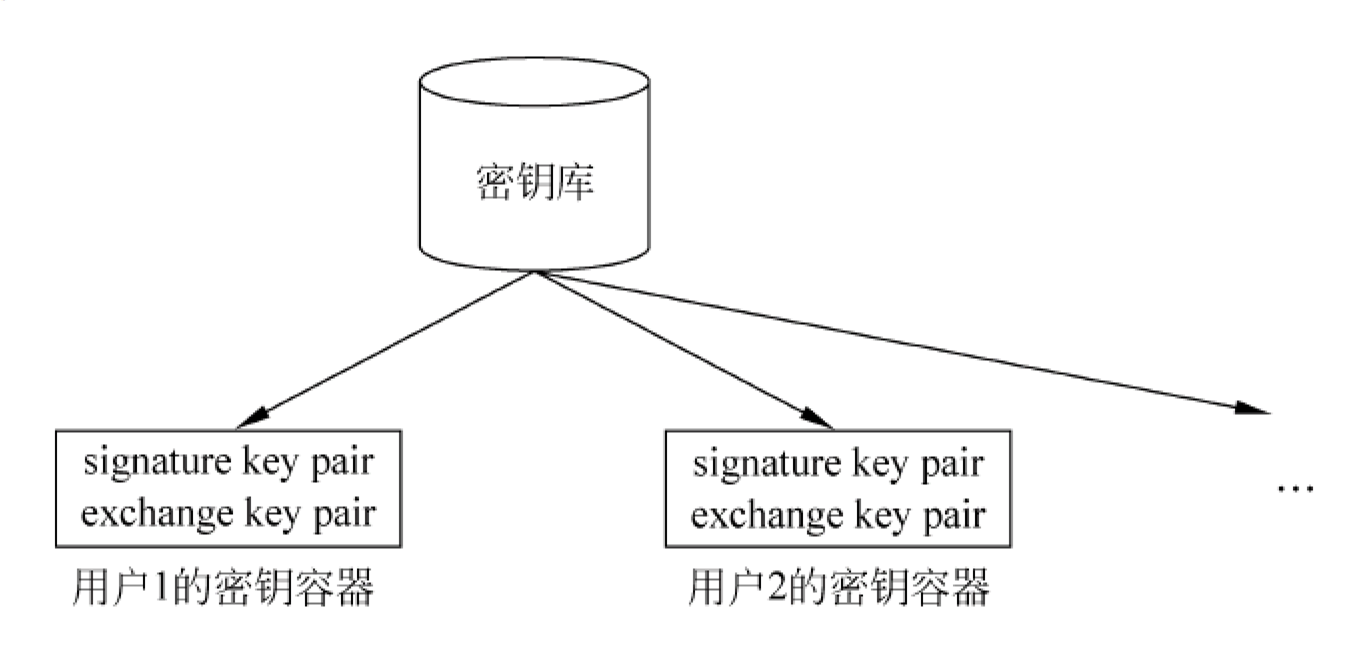


图 9-3 密钥库的构成

CSP 密钥库的逻辑结构类似于硬盘的逻辑分区,如果做一个类比,CSP 类似于一块硬盘,硬盘的各个逻辑分区,如逻辑盘 C、D、E 等,分别对应于一个密钥容器,而各逻辑分区中存储的数据则对应着密钥容器中存储的密钥。

9.1.4 微软提供的 CSP

目前,微软公司提供了 10 种 CSP,表 9-2 说明了各种 CSP 的类型和唯一名称。

表 9-2 微软公司提供的 CSP

CSP 服务名称	类　型	唯一名称
Microsoft Base Cryptographic Provider	PROV_RSA_FULL	MS_DEF_PROV
Microsoft Enhanced Cryptographic Provider	PROV_RSA_FULL	MS_ENHANCED_PROV
Microsoft Strong Cryptographic Provider	PROV_RSA_FULL	MS_STRONG_PROV
Microsoft AES Cryptographic Provider	PROV_RSA_AES	MS_ENH_RSA_AES_PROV
Microsoft DSS Cryptographic Provider	PROV_DSS	MS_DEF_DSS_PROV
Microsoft Base DSS and Diffie-Hellman Cryptographic Provider	PROV_DSS_DH	MS_DEF_DSS_DH_PROV
Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider	PROV_DSS_DH	MS_ENH_DSS_DH_PROV
Microsoft DSS and Diffie-Hellman/Schannel Cryptographic Provider	PROV_DH_SCHANNEL	MS_DEF_DH_SCHANNEL_PROV
Microsoft RSA/Schannel Cryptographic Provider	PROV_RSA_SCHANNEL	MS_DEF_RSA_SCHANNEL_PROV
Microsoft RSA Signature Cryptographic Provider	PROV_RSA_SIG	MS_DEF_RSA_SIG_PROV

下面对各种 CSP 的特点进行简要的介绍。

Microsoft Base Cryptographic Provider 是微软最早提供的一个基础版本的 CSP,其在 CryptoAPI 1.0 和 2.0 版本中均提供服务。它针对绝大多数的用途,支持数字签名和数据加密服务。所有公钥操作均支持 RSA 公钥算法。该 CSP 为默认情况下的 CSP。

Microsoft Enhanced Cryptographic Provider 针对 Microsoft Base Cryptographic Provider 安全性的不足，增加了密钥长度并且提供了额外的安全算法，增强了 CSP 的安全性。

Microsoft Strong Cryptographic Provider 是采用完全 RSA 的 CSP，它支持 Microsoft Enhanced Cryptographic Provider 的所有算法和相同的密钥长度。表 9-3 对前三种 CSP 的区别进行了说明，与 Microsoft Base Cryptographic Provider 相比，其他两种 CSP 通过增加签名、密钥交换和加密算法的密钥长度，提高了安全性，同时加入了对双密钥 TDES 和三密钥 TDES 加密算法的支持。

表 9-3 Base，Enhanced 和 Strong 三种 CSP 的区别 （单位：bits）

	RSA 签名	RSA 密钥交换	RC2	RC4	DES	TDES（2KEY）	TDES（3KEY）
Base	512	512	40	40	56	不支持	不支持
Enhanced	1024	1024	128	128	56	112	168
Strong	1024	1024	128	128	56	112	168

Microsoft AES Cryptographic Provider 加入对 AES 算法的支持，它支持 128 位、192 位和 256 位的 AES 算法。与 Microsoft Base Cryptographic Provider 相比，增强了密钥长度，其支持的密钥长度与 Microsoft Enhanced Cryptographic Provider 的相同。

Microsoft DSS Cryptographic Provider 支持 Hash 运算、数据签名、采用 SHA 和数字签名标准(DSS)的签名认证。

Microsoft Base DSS and Diffie-Hellman Cryptographic Provider 支持 Diffie-Hellman 密钥交换、SHA 运算、DSS 数据签名和认证。

Microsoft Enhanced DSS and Diffie-Hellman Cryptographic Provider 支持 Diffie-Hellman 密钥交换、SHA 运算、DSA 数据签名和认证、RC4 加密。

Microsoft DSS and Diffie-Hellman/Schannel Cryptographic Provider 支持 Hash 运算，DSS 数据签名，产生、交换和导出 Diffie-Hellman 密钥。该 CSP 支持采用 SSL3 和 TLS1 协议的密钥分散。

Microsoft RSA/Schannel Cryptographic Provider 支持 Hash 运算，数据签名和签名认证。CALG_SSL3_SHAMD5 算法用于 SSL3.0 和 TLS1.0 协议的客户端认证，Hash 值由 MD5 算法的结果和经过私钥签名的 SHA 运算结果拼接而成。该 CSP 支持 SSL2、PCT1、SSL3 和 TLS1 协议的密钥分散。

Microsoft RSA Signature Cryptographic Provider 提供数据签名和认证操作。

9.2 CryptoAPI 介绍

微软 CryptoAPI 是微软操作系统提供的实现加密运算功能的上层应用开发接口。CryptoAPI 目前有两个版本：1.0 和 2.0，当前版本是 2.0。CryptoAPI 1.0 包括基本加解密函数，这些函数提供安全通信的保密性和数据的完整性，但是没有提供身份认证功能。CryptoAPI 2.0 在 CrytoAPI 1.0 的基础上，增加了身份认证功能，CryptoAPI 2.0 的身份认证功能是通过数字证书实现的。CryptoAPI 的体系结构如图 9-4 所示。

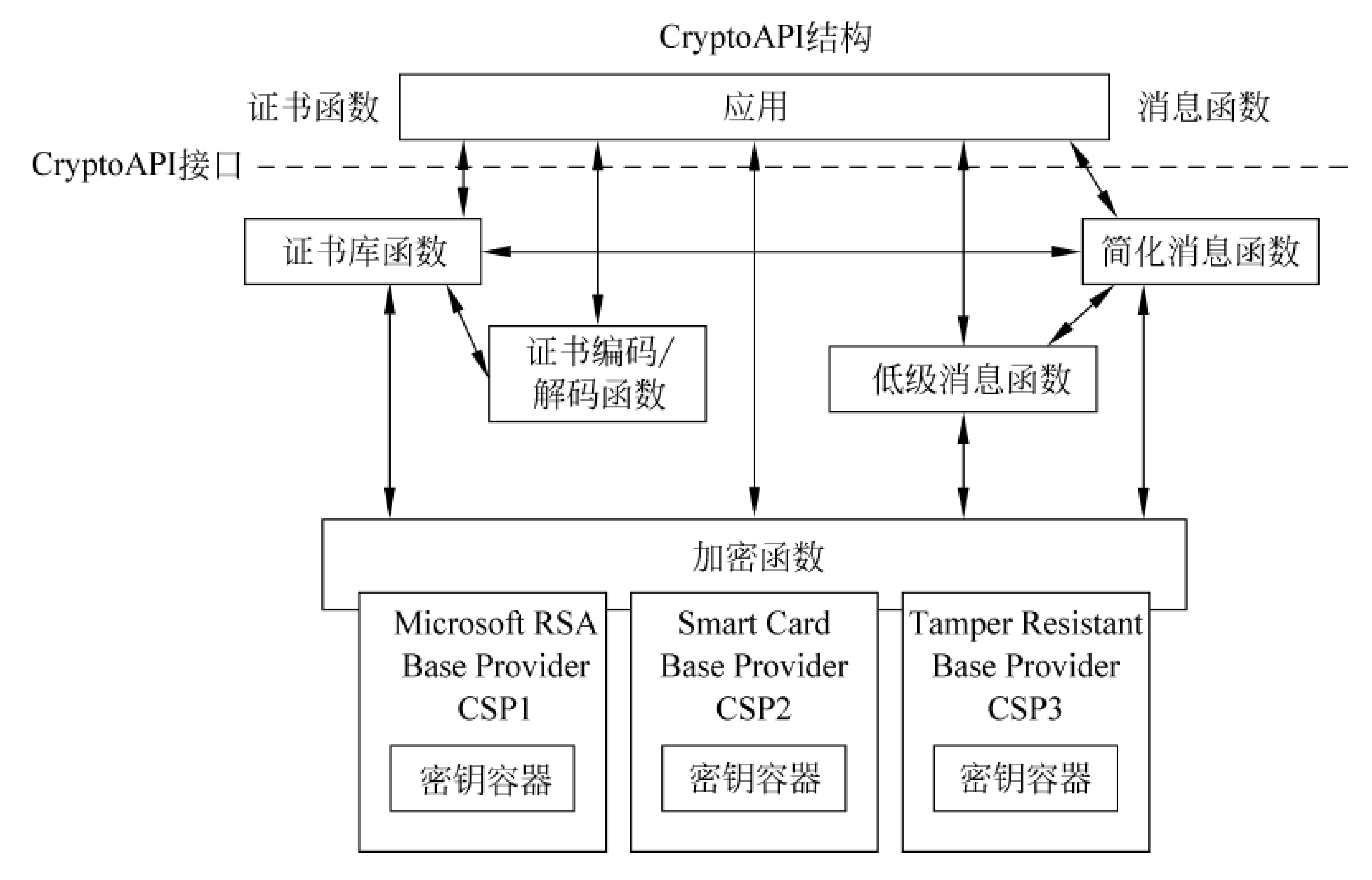


图 9-4　CryptoAPI 的体系结构

CryptoAPI 提供了很多函数，包括编码、解码、加密、解密、Hash、数字证书、证书管理和证书存储等功能。对于加密和解密，CryptoAPI 同时提供基于会话密钥和公私钥对的加密方法。CryptoAPI 提供接口函数可分为 5 类：基本加密函数、证书和证书库函数、证书验证函数、消息函数和辅助函数，如表 9-4 所示。

表 9-4　CryptoAPI 函数简介

基本加密函数	服务提供者函数
	密钥产生和交换函数
	编码/解码函数
	数据加密/解密函数
	哈希和数字签名函数
证书和证书库函数	证书库函数
	维护函数
	证书函数
	证书撤销列表函数
	证书信任列表函数
	扩展属性函数
证书验证函数	使用 CTL 的函数
	证书链验证函数
消息函数	低级消息函数
	简化消息函数
辅助函数	数据管理函数
	数据转换函数
	增强密钥用法函数
	密钥标识函数
	证书库回调函数
	OID 支持函数
	远程对象恢复函数
	PFX 函数

9.2.1 基本加密函数

基本加密函数为开发加密应用程序提供了基本的服务，所有与 CSP 的通信都是通过基本加密函数实现的。基本加密函数分为 5 种：服务提供者函数、密钥的产生和交换函数、编码/解码函数、数据加密/解密函数、哈希和数字签名函数。

1. 服务提供者函数

服务提供者函数提供了一组针对 CSP 的函数，提供了如连接和断开 CSP，设置和获取 CSP 信息等功能，函数描述如表 9-5 所示。

表 9-5 服务提供者 API

API	说　明
CryptAcquireContext	获得指定 CSP 的密钥容器的句柄，同时获得已经存在或创建新的密钥容器
CryptContextAddRef	增加 HCRYPTPROV 句柄一个应用计数。该函数用于将该句柄作为结构体中的一个成员发送给其他函数的应用。调用该函数后，需相应增加调用 CryptReleaseContext 的次数
CryptEnumProviders	枚举当前计算机中的 CSP，获得如 CSP 名称和类型等信息
CryptEnumProviderTypes	枚举当前计算机中 CSP 的类型信息
CryptGetDefaultProvider	获得当前用户或计算机的默认 CSP 的名称
CryptGetProvParam	获得指定 CSP 的参数，如当前密钥容器名称、CSP 名称等信息
CryptInstallDefaultContext	安装先前得到的 HCRYPTPROV 上下文作为当前默认的上下文
CryptReleaseContext	释放由 CryptAcquireContext 得到的句柄，调用该函数一次，句柄应用计数减一，当减为零时，应用程序将无法使用该句柄
CryptSetProvider 和 CryptSetProviderEx	为当前用户或当前计算机指定默认 CSP
CryptSetProvParam	设置 CSP 的参数
CryptUninstallDefaultContext	该函数删除先前由 CryptInstallDefaultContext 函数安装的默认上下文

2. 密钥产生和交换函数

密钥产生和交换函数是一组针对密钥操作的函数。它提供了密钥获取、产生、导入、导出、设置、销毁等功能，函数描述如表 9-6 所示。

表 9-6 密钥产生和交换 API

API	说　明
CryptAcquireCertificatePrivateKey	获得指定证书的私钥。该函数用于已经得到用户证书，但没有得到 CSP 句柄的场合。该函数得到一个 CSP 句柄 HCRYPTPROV 和指向私钥的指针 dwKeySpec
CryptDeriveKey	从一个指定的种子派生出加密会话密钥
CryptDestoryKey	释放密钥句柄。密钥句柄释放后，则不能够再使用
CryptDuplicateKey	复制一个密钥及其状态。该函数通常应用在使用只有 Salt Value 不同的密钥加密多个消息的场合
CryptExportKey	将 CSP 中存储的密钥，以 BLOB 的格式导出
CryptGenKey	产生 CSP 应用的一个随机会话密钥或公私密钥对
CryptGenRandom	产生一个随机数
CryptGetKeyParam	得到密钥的参数

续表

API	说　明
CryptGetUserKey	得到一个交换密钥或签名密钥的句柄
CryptImportKey	把一个密钥 BLOB 导入 CSP 中
CryptSetKeyParam	设置密钥参数

3. 编码/解码函数

编码/解码函数用于编码和解码证书、证书撤销列表、证书请求和证书扩展。函数描述如表 9-7 所示。

表 9-7　编码/解码 API

API	说　明
CryptDecodeObject	对指定类型的结构体进行解码
CryptDecodeObjectEx	对指定类型的结构体进行解码,此函数支持内存分配选项
CryptEncodeObject	对指定类型的结构体进行编码
CyptEncodeObjectEx	对指定类型的结构体进行编码,此函数支持内存分配选项

4. 数据加密/解密函数

数据加密/解密函数支持加密和解密操作,函数描述如表 9-8 所示。CryptEncrypt 和 CryptDecrypt 函数调用前需要指定密钥,该密钥可由 CryptGenKey、CryptDeriveKey 或 CryptImportKey 产生。产生密钥的同时也制定了加密算法,CryptSetKeyParam 可以指定额外的加密参数。

表 9-8　数据加密/解密 API

API	说　明
CryptDecrypt	使用指定加密密钥来解密一段密文
CryptEncrypt	使用指定加密密钥来加密一段明文
CryptProtectData	对 DATA_BLOB 结构进行加密
CryptUnprotectData	对 DATA_BLOB 结构的完整性进行验证和解密

5. 哈希和数字签名函数

哈希和数字签名函数用于计算 Hash 值、计算和验证数字签名。函数描述如表 9-9 所示。

表 9-9　Hash 和数字签名 API

API	说　明
CryptCreateHash	创建一个空 Hash 对象
CryptDestoryHash	销毁一个 Hash 对象
CryptDuplicateHash	复制一个 Hash 对象
CryptGetHashParam	得到一个 Hash 对象参数
CryptHashData	对一块数据进行 Hash 运算,把 Hash 值加到指定的 Hash 对象中
CryptHashSessionKey	对一个会话密钥进行 Hash 运算,把 Hash 值加到指定的 Hash 对象中
CryptSetHashParam	设置 Hash 对象的参数
CryptSignHash	对一个 Hash 对象进行签名
CryptVerifySignature	对一个数字签名进行验证

9.2.2 证书和证书库函数

证书和证书库函数负责证书、证书撤销列表和证书信任列表的使用、存储和获取工作。这些函数可以分成6组：证书库函数、证书维护函数、证书函数、证书撤销列表函数、证书信任列表函数和扩展属性函数。

1. 证书库函数

一个用户可以拥有多个证书，通常，这些证书包括该用户使用的证书和与其他个人和实体进行通信的证书。每个实体的证书可能多于一个。每个个人证书应该有一个可以回溯到信任根证书的证书链。证书库函数提供了存储、获取、枚举、验证和使用证书等操作。函数描述如表9-10所示。

表 9-10 证书库 API

API	说明
CertAddStoreToCollection	在证书库中增加一个证书
CertCloseStore	关闭一个证书库句柄
CertControlStore	如果缓冲区中证书和证书库中证书本身内容不相符时，允许给应用程序发一个通知
CertDuplicateStore	通过增加引用计数来复制证书库句柄
CertEnumPhysicalStore	对于指定系统库枚举物理库
CertEnumSystemStore	枚举所有可用的系统库
CertEnumSystemStoreLocation	枚举可用系统库的所有位置
CertGetStoreProperty	得到一个库的属性
CertOpenStore	使用指定库类型来打开证书库
CertOpenSystemStore	打开一个系统证书库
CertRegisterPhysicalStore	在一个注册系统库里增加一个物理库
CertRegisterSystemStore	注册一个系统库
CertRemoveStoreFromCollection	从一个库集合里删除证书库
CertSaveStore	保存证书库
CertSetStoreProperty	设置证书属性
CertUnregisterPhysicalStore	从系统库中删除一个物理库
CertUnregisterSystemStore	反注册一个指定系统库

2. 维护函数

维护函数提供对证书和证书库的维护。函数描述如表9-11所示。

表 9-11 维护函数 API

API	说明
CertAddSerializeElementToStore	向库中添加可串行化的证书、证书撤销列表和证书信任列表
CertCreateContext	从编码字节中创建指定上下文
CertEnumSubjectInSortedCTL	在存储的证书信任列表中枚举信任主体
CertFindSubjectInCTL	在证书信任列表库中寻找指定的主体
CertFindSubjectInSortedCTL	在存储的证书信任列表中寻找指定的主体

3. 证书函数

绝大部分证书函数与证书信任列表和证书撤销列表的处理相关。函数描述如表 9-12 所示。

表 9-12 证书 API

API	说　明
CertAddCertificateContextToStore	在证书库里增加一个证书上下文
CertAddCertificateLinkToStore	在证书库里增加一个链接，该链接指向不同库里的证书上下文
CertAddEncodedCertificateToStore	把编码证书转换成证书上下文并且把它添加到证书库里
CertCreateCertificateContext	从编码证书中创建一个证书上下文
CertCreateSelfSignCertificate	创建一个自签名证书
CertDeleteCertificateFromStore	从证书库里删除一个证书
CertDuplicateCertificate	通过增加引用计数来复制证书上下文
CertEnumCertificateInStore	在证书库里枚举证书上下文
CertFindCertificateInStore	在证书库里寻找证书上下文
CertFreeCertificateContext	释放一个证书上下文
CertGetIssuerCertificateFromStore	在证书库里得到指定主体证书的发行者
CertGetSubjectCertificateFromStore	获得主体证书的上下文
CertGetValidUsages	获得证书的用法
CertSerializeCertificateStoreElement	串行化编码证书的证书上下文
CertVerifySubjectCertificateContext	使用发行者来验证主体证书
CryptUIDlgViewContext	显示证书、证书信任列表或证书撤销列表信息
CryptUIDlgSelectCertificateFromStore	显示对话框，用户可从指定库中选择证书

4. 证书撤销列表函数

提供一组证书撤销列表存储和获取的函数。函数描述如表 9-13 所示。

表 9-13 证书撤销列表 API

API	说　明
CertAddCRLContextToStore	在证书库里增加一个 CRL 上下文
CertAddCRLLinkToStore	在不同的库里增加一个 CRL 上下文链接
CertAddEncodedCRLToStore	从编码 CRL 创建 CRL 上下文，然后把它加入到证书库中
CertCreateCRLContext	从编码 CRL 创建 CRL 句柄，但不把它添加到库中
CertDeleteCRLFromStore	从证书库里删除一个 CRL
CertDuplicateCRLContext	通过增加引用计数来复制 CRL 上下文
CertEnumCRLsInStore	枚举证书库里的 CRL 句柄
CertFindCertificateInCRL	依据指定证书寻找 CRL 列表
CertFindCRLInStore	在库里寻找 CRL 上下文
CertFreeCRLContext	释放 CRL 上下文
CertGetCRLFromStore	从库里得到 CRL 上下文句柄
CertSerializeCRLStoreElement	串行化编码的 CRL 上下文和编码后的属性

5. 证书信任列表函数

提供一组证书信任列表存储和获取的函数。函数描述如表 9-14 所示。

表 9-14 证书信任列表 API

API	说　明
CertAddCTLContextToStore	把一个 CTL 上下文加入到证书库里
CertAddCTLLinkToStore	给不同库里的 CTL 上下文添加链接
CertAddEncodedCTLToStore	从编码 CTL 创建 CTL,并且把它加到证书库里
CertCreateCTLContext	从编码 CTL 创建 CTL 上下文
CertDeleteCTLFromStore	从证书库里删除 CTL
CertDuplicateCTLContext	通过增加引用计数来复制 CTL 上下文
CertEnumCTLsInStore	在证书库里枚举 CTL 上下文
CertFindCTLInStore	在证书库里查找 CTL 上下文
CertFreeCTLContext	释放 CTL 上下文
CertSerializeCTLStoreElement	串行化编码的 CTL 上下文和编码后的属性

6. 扩展属性函数

提供一组与证书、证书撤销列表和证书信任列表的扩展属性相关的函数。函数描述如表 9-15 所示。

表 9-15 扩展属性 API

AP	说　明
CertEnumCertificateContextProperties	枚举指定证书上下文的属性
CertEnumCRLContextProperties	枚举指定 CRL 上下文的属性
CertEnumCTLContextProperties	枚举指定 CTL 上下文的属性
CertGetCertificateContextProperty	得到证书属性
CertGetCRLContextProperty	得到 CRL 属性
CertGetCTLContextProperty	得到 CTL 属性
CertSetCertificateContextProperty	设置证书属性
CertSetCRLContextProperty	设置 CRL 属性
CertSetCTLContextProperty	设置 CTL 属性

9.2.3 证书验证函数

证书验证函数提供采用证书信任列表和证书链验证两种方式。

1. 使用 CTL 验证的函数

提供了一组采用 CTL 验证证书的函数。函数描述如表 9-16 所示。

表 9-16 使用 CTL 的 API

API	说　明
CertVerifyCTLUsage	验证一个主体针对某个用法是可信任的,通过在包括该主体并且含有用法标识的 CTL 中查找,该 CTL 必须是经签名的且时间有效
CryptMsgEncodeAndSignCTL	编码 CTL 并且创建一个包含编码 CTL 的签名消息
CryptMsgGetAndVerifySigner	验证加密消息的签名
CryptMsgSignCTL	创建包含编码 CTL 的签名消息

2. 证书链验证函数

证书链用来对单一证书提供信任信息。函数描述如表 9-17 所示。

表 9-17 证书链验证 API

API	说　　明
CertCreateCertificateChainEngine	为应用程序创建一个新的非缺省的链引擎
CertCreateCTLEntryFromCertificateContextProperties	创建一个 CTL 入口
CertDuplicateCertificateChain	通过增加引用计数来复制证书链
CertFindChainInStore	在证书库里查找证书链
CertFreeCertificateChain	释放证书链
CertFreeCertificateChainEngine	释放证书链引擎
CertGetCertificateChain	从最后一个证书建立一个上下文链表
CertSetCertificateContextPropertiesFromCTLEntry	用 CTL 入口属性来设置证书上下文的属性
CertIsValidCRLForCertificate	通过检查 CRL 来确定其是否包括指定被撤销的证书
CertVerifyCertificateChainPolicy	通过检查证书链来确定其有效性

9.2.4 消息函数

消息函数由低级消息函数和简化消息函数两部分组成。低级消息函数直接处理 PKCS#7 消息。这些函数编码用于传输的 PKCS#7 数据,解码收到的 PKCS#7 数据,同时对收到的数据进行解密和签名验证。简化消息函数在更高的层次上,将多个低级消息函数和证书函数包含到一个函数中,来实现一定的功能。简化消息函数减少了调用函数的次数,简化了 CryptoAPI 的应用。

1. 低级消息函数

低级消息函数直接处理 PKCS#7 消息,函数描述如表 9-18 所示。这些函数编码用于传输的 PKCS#7 数据,解码收到的 PKCS#7 数据,同时对收到的数据进行解密和签名验证。在大多数应用程序中,并不建议使用低级消息函数,建议使用简化消息函数。

表 9-18 低级消息 API

API	说　　明
CryptMsgCalculateEncodedLength	计算加密消息的长度
CryptMsgClose	关闭加密消息的句柄
CryptMsgControl	执行指定的控制函数
CryptMsgCountersign	标记消息中已存在的签名
CryptMsgCountersignEncoded	标记已存在的签名
CryptMsgDuplicate	通过增加引用计数来复制加密消息句柄
CryptMsgGetParam	得到加密消息编码或者解码后的参数
CryptMsgOpenToDecode	打开加密消息进行解码
CryptMsgOpenToEncode	打开加密消息进行编码
CryptMsgUpdate	更新加密消息的内容
CryptMsgVerifyCountersignatureEncoded	验证 SignerInfo 结构中标记时间
CryptMsgVerifyCountersignatureEncodedEx	验证 SignerInfo 结构中标记时间,签名者可以是 CERT_PUBLIC_KEY_INFO 结构、证书上下文或证书链上下文

2. 简化消息函数

简化消息函数将多个低级消息函数包含到一个函数中，用来实现一定的功能。函数描述如表 9-19 所示。

表 9-19 简化消息 API

API	说　明
CryptDecodeMessage	对加密消息进行解码
CryptDecryptAndVerifyMessageSignature	对指定消息进行解密并且验证签名者
CryptDecryptMessage	解密指定消息
CryptEncryptMessage	加密指定消息
CryptGetMessageCertificates	返回包含消息的证书和 CRL 的证书库
CryptGetMessageSignatureCount	返回签名消息的签名者数量
CryptHashMessage	创建消息的 Hash
CryptSignAndEncryptMessage	对消息进行签名并且加密
CryptSignMessage	对消息进行签名
CryptVerifyDetachedMessageHash	验证包含已解绑定 Hash 的 Hash 消息
CryptVerifyDetachedMessageSignature	验证包含已解绑定签名的签名消息
CryptVerifyMessageHash	验证一个 Hash 消息
CryptVerifyMessageSignature	验证一个签名消息

9.2.5 辅助函数

辅助函数分为 8 部分：数据管理函数、数据转换函数、增强密钥用法函数、密钥标识函数、证书库回调函数、OID 支持函数、远程对象恢复函数、PFX 函数。

1. 数据管理函数

提供一组用于管理数据和证书的函数。函数描述如表 9-20 所示。

表 9-20 数据管理 API

API	说　明
CertCompareCertificate	比较两个证书是否相同
CertCompareCertificateName	比较两个证书名称是否相同
CertCompareIntegerBlob	比较两个整数 BLOB
CertComparePublicKeyInfo	比较两个证书公钥是否相同
CertFindAttribute	通过 OID 来查找属性
CertFindExtension	通过 OID 来查找扩展
CertFindRDNAttr	通过 OID 来查找相对不同名称(relative distinguished name,RDN)属性
CertGetIntendedKeyUsage	从证书中取得相关密钥用法
CertGetPublicKeyLength	从公钥 BLOB 中取得公钥/私钥长度
CertIsRDNAttrsInCertificateName	通过指定 RDN 数组属性比较证书名称属性来决定证书是否已包含了所有属性
CertVerifyCRLRevocation	验证主体证书是否在 CRL 中
CertVerifyCRLTimeValidity	验证 CRL 的有效时间
CertVerifyRevocation	验证主题证书是否在 CRL 中

续表

API	说　明
CertVerifyTimeValidity	验证证书的有效时间
CertVerifyValidityNesting	验证主体时间的有效性是否在发行者有效时间内
CryptExportPKCS8	该函数已由 CryptExportPKCS8Ex 取代
CryptExportPKCS8Ex	以 PKCS#8 的格式导出私钥
CryptExportPublicKeyInfo	导出公钥信息
CryptExportPublicKeyInfoEx	导出公钥信息(用户可以指定算法)
CryptFindCertificateKeyProvInfo	枚举 CSP 和它的密钥容器来查找对应于公钥的相应私钥
CryptFindLocalizedName	查找指定名称的局部化名称
CryptHashCertificate	Hash 证书内容
CryptHashPublicKeyInfo	计算公钥信息的 Hash
CryptHashToBeSigned	计算签名内容的 Hash 值
CryptImportPkCS8	将私钥从 PKCS8 转换,并且导入到提供者
CryptImportPublicKeyInfo	把公钥信息导入 CSP 并且返回它的句柄
CryptImportPublicKeyInfoEx	把公钥信息导入 CSP 并且返回它的句柄
CryptMemAlloc	分配内存,用于 Crypt32.lib 函数使用
CryptMemFree	释放由 CryptMemAlloc 或 CryptMemRealloc 分配的内存
CryptMemRealloc	释放已经分配的内存,重新分配内存
CryptQueryObject	得到 BLOB 或文件的内容信息
CryptSignAndEncodeCertificate	编码需要签名的信息,签名编码信息,编码签名消息,编码信息
CryptSignCertificate	对编码内容中的待签名信息进行签名
CryptSIPAddProvider	加入一个主体接口包(SIP)
CryptSIPRemoveProvider	删除由 CryptSIPAddProvider 添加的 SIP
CryptVerifyCertificateSignature	使用公钥信息对主体证书或 CRL 的签名进行验证
CryptVerifyCertificateSignatureEx	使用公钥信息对主体证书或 CRL 的签名进行验证
GetEncSChannel	在内存中存储加密 Schannel DLL 内容

2. 数据转换函数

提供一组将证书中的成员变量转换为其他形式的函数。函数描述如表 9-21 所示。

表 9-21　数据转换 API

API	说　明
CertAlgIdToOID	把 CSP 算法标识符转换成 OID
CertGetNameString	得到证书的主题或颁发者名称并且把它转换成字符串
CertNameToStr	把证书名称由 CERT_NAME_BLOB 格式转换成字符串
CertOIDToAlgId	把 OID 转换成 CSP 算法标识符
CertRDNValueToStr	把名称从 CERT_RDN_VALUE_BLOB 结构转换成字符串
CertStrToName	把字符串转换成编码证书名称
CryptBinaryToString	把二进制序列转换成字符串
CryptFormatObject	格式化编码数据,返回 Unicode 字符串
CryptStringToBinary	把格式化的字符串转换成二进制序列

3. 增强密钥用法函数

提供了一组用于处理增强密钥用法(enhanced key usage,EKU)扩展和证书 EKU 扩展特性的函数。函数描述如表 9-22 所示。

表 9-22 增强密钥用法 API

API	说　明
CertAddEnhancedKeyUsageIdentifier	在证书 EKU 属性中增加一个用法标识符 OID
CertGetEnhancedKeyUsage	获得证书的 EKU 扩展或属性信息
CertRemoveEnhancedKeyUsageIdentifier	从证书 EKU 扩展属性中删除用法标识符 OID
CertSetEnhancedKeyUsage	设置证书的 EKU 属性

4. 密钥标识函数

密钥标识函数允许用户创建、设置、重新获取、定位一个密钥标识或其属性,函数描述如表 9-23 所示。密钥标识是公私密钥对的唯一标识,通常是编码后的 CER_PUBLIC_KEY_INFO 的 20 字节 SHA1hash 值。从证书的 CER_KEY_INDENTIFIER_PROP_ID 可以获得密钥标识。应用密钥标识可直接加密和解密消息,而不用数字证书。

表 9-23 密钥标识 API

API	说　明
CryptCreateKeyIdentifierFromCSP	从 CSP 公钥 BLOB 创建密钥标识
CryptEnumKeyIdentifierProperties	枚举密钥标识和其属性
CryptGetKeyIdentifierProperty	从指定密钥标识中获得指定属性
CryptSetKeyIdentifierProperty	设置指定密钥标识的属性

5. 证书库回调函数

提供了一组用于注册或安装应用定义的证书库和相关功能的回调函数。函数描述如表 9-24 所示。

表 9-24 证书库回调 API

API	说　明
CertDllOpenStoreProv	定义证书提供者打开函数
CertStoreProvCloseCallback	确定当打开证书库索引计数器为零时如何操作
CertStoreProvDeleteCertCallback	确定证书从证书库中删除前将采取何种操作
CertStoreProvDeleteCRLCallback	确定 CRL 从证书库中删除前将采取何种操作
CertStoreProvReadCertCallback	目前没有使用,可能将来在 CSP 上使用
CertStoreProvReadCRLCallback	目前没有使用,可能将来在 CSP 上使用
CertStoreProvSetCertPropertyCallback	确定在调用 CertSetCertificateContextProperty 和 CertGetCertificate-ContextProperty 函数之前采用何种操作
CertStoreProvSetCRLPropertyCallback	确定在调用 CertSetCRLContextProperty 和 CertGetCRLContext-Property 函数之前采用何种操作
CertStoreProvWriteCertCallback	确定向证书库中增加证书前采取何种操作
CertStoreProvWriteCRLCallback	确定向证书库中增加 CRL 前采取何种操作

续表

API	说　明
CertStoreProvReadCTL	读取提供者的 CTL 上下文，如果该 CTL 存在，则创建一个新的 CTL 上下文
CertStoreProvWriteCTL	确定 CTL 是否能够加入到证书库中
CertStoreProvDeleteCTL	确定是否能够删除 CTL
CertStoreProvSetCTLProperty	确定是否能够配置 CTL 的某个属性
CertStoreProvControl	当存储在高速缓冲存储器中的证书库内容与存储在永久存储器中的证书库内容不同时，通知应用程序
CertStoreProvFindCert	在证书库中，查找满足指定标准的第一个或下一个证书
CertStoreProvFreeFindCert	释放先前找到的证书上下文
CertStoreProvGetCertProperty	重新获取证书的特定属性
CertStoreProvFindCRL	在证书库中，查找满足指定标准的第一个或下一个 CRL
CertStoreProvFreeFindCRL	释放先前找到的 CRL 上下文
CertStoreProvGetCRLProperty	重新获取 CRL 的特定属性
CertStoreProvFindCTL	在证书库中，查找满足指定标准的第一个或下一个 CTL
CertStoreProvFreeFindCTL	释放先前找到的 CTL 上下文
CertStoreProvGetCTLProperty	重新获取 CTL 的特定属性

6. OID 支持函数

提供了一组支持对象标识符(OID)的函数。这些函数可以安装、注册、分发 OID 和指定类型的函数。函数描述如表 9-25 所示。

表 9-25　OID 支持 API

API	说　明
CryptEnumOIDFuction	枚举已经注册的 OID 函数
CryptEnumOIDInfo	枚举预先定义或已经注册的 OID 信息
CryptFindOIDInfo	使用指定的密钥类型和密钥查找 OID 信息
CryptFreeOIDFuctionAddress	释放 OID 函数地址句柄
CryptGetDefaultOIDDllList	对于指定的函数集合和编码类型获得默认注册的 DLL 入口列表
CryptGetDefaultOIDFuctionAddress	获得已安装的第一个或下一个默认函数或者加载包含默认函数的 DLL
CryptGetOIDFuctionAddress	搜索匹配指定编码类型和 OID 的安装函数列表，如果没有找到，就查找注册表
CryptGetOIDFuctionValue	获得指定编码类型、函数名称、OID、变量名称的值
CryptInitOIDFuctionSet	依据提供的函数名，初始化和返回 OID 函数集合的句柄
CryptInstallOIDFuctionAddress	安装可调用的 OID 函数地址集合
CryptRegisterDefaultOIDFuction	注册包含指定编码类型和函数名称的默认函数的 DLL
CryptRegisterOIDFuction	注册包含指定编码类型、函数名称和 OID 函数的 DLL
CryptRegisterOIDInfo	注册由 CRYPT_OID_INFO 指定的 OID 信息，并将其写入注册表
CryptSetOIDFuctionValue	设置指定编码类型、函数名称、OID、变量名称的值
CryptUnregisterDefaultOIDFunction	移除包含指定编码类型和函数名称的默认函数 DLL 的注册信息
CryptUnregisterOIDFuction	移除包含指定编码类型、函数名称和 OID 的函数 DLL 的注册信息
CryptUnregisterOIDInfo	移除指定 OID 的注册信息

7. 远程对象恢复函数

支持用户恢复 PKI 对象、证书 URL、CTL、CRL，或者从一个对象提取 URL。函数描述如表 9-26 所示。

表 9-26　远程对象恢复 API

API	说　　明
CryptGetObjectUrl	从证书、CTL 或 CRL 中取得远程对象的 URL
CryptRetrieveObjectByUrl	由 URL 指定位置恢复 PKI 对象

8. PFX 函数

支持个人信息交换(personal information exchange，PFX)格式的 BLOB。函数描述如表 9-27 所示。

表 9-27　PFX API

API	说　　明
PFXExportCertStore	从证书库中导出证书或相关的私钥，该函数是与 Internet Explorer 4.0 客户端兼容的旧版本函数
PFXExportCertStoreEx	从证书库中导出证书或相关的私钥，该函数替代了旧版本的 PFXExportCertStore 函数
PFXImportCertStore	导入 PFX BLOB，并且返回包含证书和相关私钥的证书库的句柄
PFXIsPFXBlob	尝试将 BLOB 外层以 PFX 包的形式解码
PFXVerifyPassword	尝试将 BLOB 外层以 PFX 包的形式解码，并且用给定的密钥解密

9.3 CryptoAPI 应用

9.3.1 *如何应用* CryptoAPI

应用程序并不直接与 CSP 进行交互，它通过调用 CryptoAPI 函数来访问 CSP。CryptoAPI 是通过 Windows 操作系统的 advapi32.dll 和 crypt32.dll 实现的。采用 Dependency Walker 软件对上述两个动态链接库文件进行分析，结果如图 9-5 和图 9-6 所示。

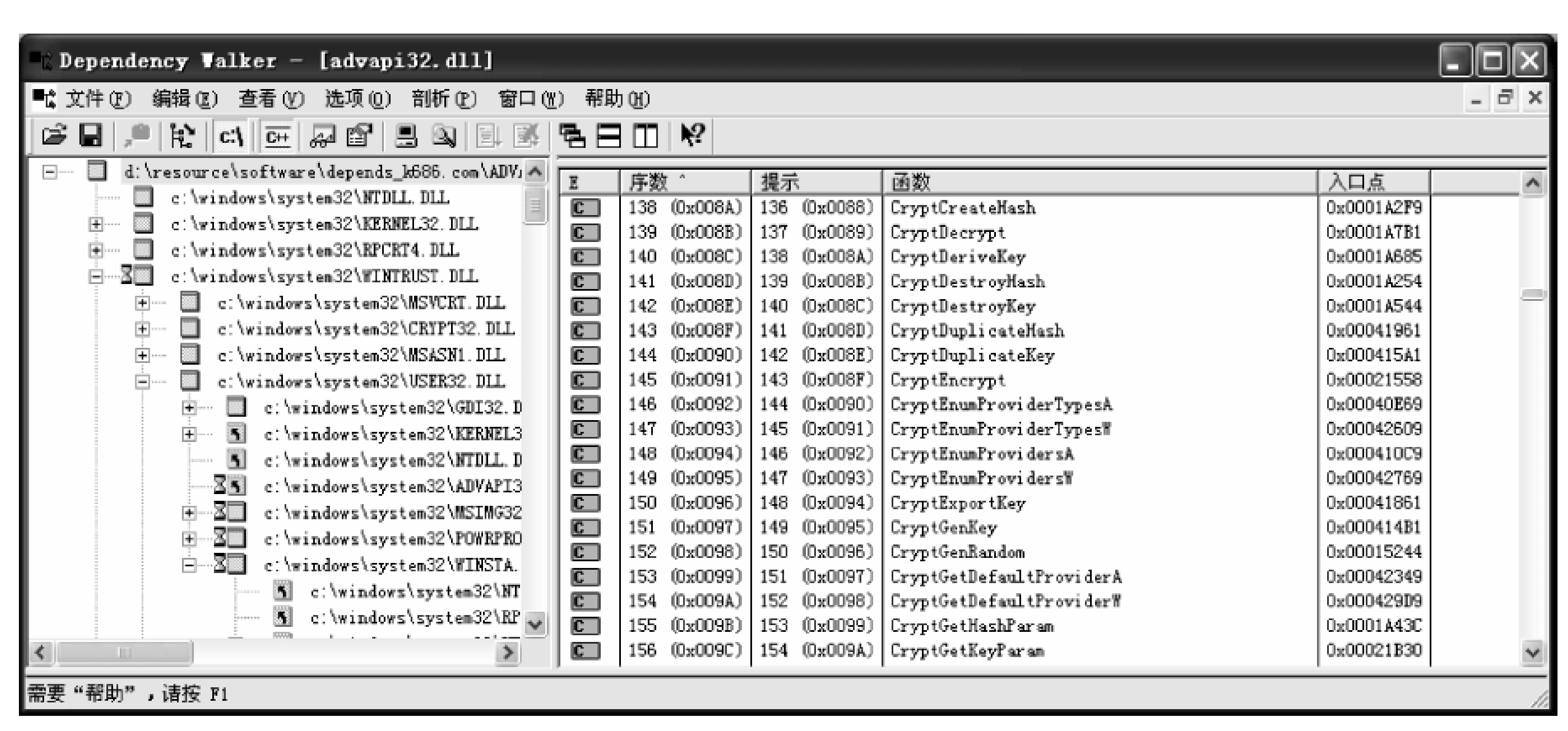


图 9-5　对 advapi32.dll 的分析结果

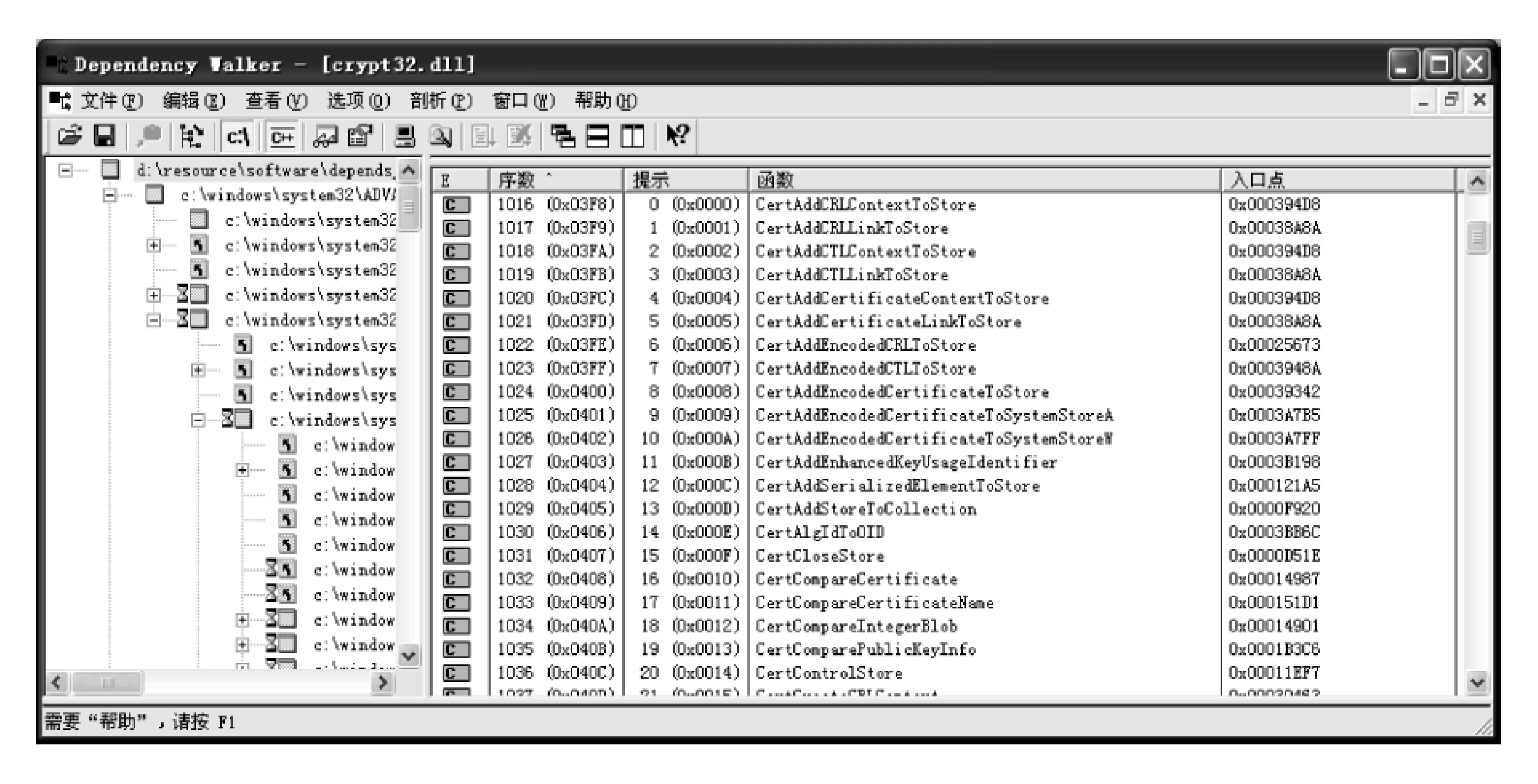


图 9-6 对 crypt32.dll 的分析结果

分析结果显示了很多 CryptoAPI 函数，应用程序就是通过这两个动态链接库提供的接口来向操作系统发出指令，请求 CSP 提供服务的。CSP 中的数据对象包括密钥容器对象、Hash 对象、密钥对象等。其中句柄对与应用层和 CSP 层是透明的，也就是说应用程序使用的句柄和 CSP 中的句柄是不同的。

CryptoAPI 是操作系统提供给应用程序的开发接口，前一部分对 API 函数的功能进行了简要的介绍。要开发出高质量的应用程序还需要有一定的编程经验，对如句柄(handle)和 CSP 中的数据对象等概念有一定的理解。

句柄是操作系统提供给用户的对象标识符，通过获得对象句柄，用户可以通过操作系统提供的各种 API 发出对对象操作的各种请求。更具体的来说，句柄是指向指针的指针，由于 Windows 操作系统采用了虚拟内存访问机制，对象在内存中的地址会经常发生变化，Windows 操作系统通过在内存中为各应用程序分配一些存储空间，专门用于记录对象在内存中的地址变化，而这些存储单元的地址不发生变化，其中记录了对象地址变化后的新地址。这样就形成了从句柄到对象指针的映射，通过句柄就可以间接访问对象。实际上，操作系统向用户提供句柄而非指针，主要考虑到直接提供对象指针可能带来的风险，用户利用指针可以尝试对对象中数据成员进行操作或者分析有关类实现的各种细节，这样不利于对象数据的安全，尤其在提供加密服务时这点则显得更为重要，例如用户可能尝试获取 CSP 密钥库中存储的用户密钥，或者尝试分析 CSP 实现的具体细节，这些操作显然是极具破坏性的。通过句柄，操作系统隔离了用户直接操作系统中定义对象的方法，用户若希望对系统中定义对象进行操作，必须从操作系统获得该对象的句柄(唯一标识号)，通过该句柄向操作系统提出各种操作的请求，操作的执行由操作系统完成，而隐藏了底层的实现细节。

CSP 中的数据对象分为三种：密钥容器对象、Hash 对象和密钥对象。其中密钥容器对象是永久对象即永久地保存在物理设备中，除非销毁密钥容器；Hash 对象和密钥对象是临时对象，仅当应用程序运行时，保存在内存中。图 9-7 说明了 CSP 与密钥容器句柄和密钥句柄之间的关系。密钥容器是 CSP 中的一个逻辑对象，一个 CSP 中可以有多个密钥容器，用户可以通过使用该对象的句柄访问其中存储的密钥。密钥存储在密钥容器中，用户可以获得密钥的句柄。

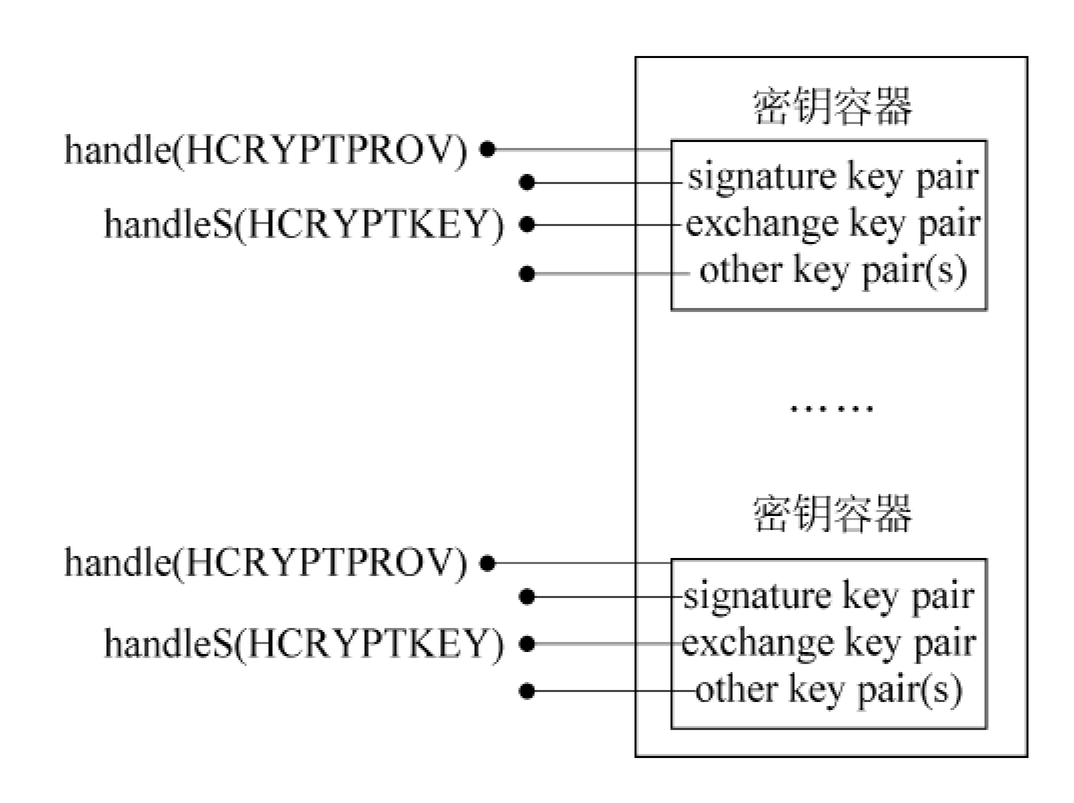


图 9-7 CSP 与密钥容器句柄和密钥句柄之间的关系

密钥容器对象中保存着密钥对,它的实现方式是灵活的,可以用软件方式实现,例如用注册表、文本文件来实现;也可以用硬件设备实现,如智能卡、USB 设备。CSP 为每个用户维护一个密钥容器。密钥容器只能保存每种类型的一个密钥,例如只能保存一个签名密钥。

Hash 对象是计算会话密钥或消息 Hash 值的对象。通过调用 CryptCreateHash 产生,调用该函数后 CSP 在其中定义该 Hash 对象。

密钥对象,也称为 Key BLOB(key binary large object),是密钥在不同 CSP 之间传送的载体。CSP 通过建立密钥库,并且在密钥库中为每个用户建立一个密钥容器来保存密钥,然而需要将密钥从 CSP 中导出时,则需要采用 Key BLOB 格式。Key BLOB 由一个固定长度的密钥头和可变长度的密钥体组成,如图 9-8 所示,通常密钥体是经过加密的。

密钥头(BLOBHEADER 结构,固定长度)	密钥体(可变长度,通常经加密处理)

图 9-8 Key BLOB 的数据结构

密钥头采用 BLOBHEADER 的数据结构,图 9-9 说明了 BLOBHEADER 的数据结构,其中密钥类型占一个字节,说明了密钥的用途,密钥类型分为四种:Public Key BLOB、Private Key BLOB、Simple BLOB 和 Symmetric Wrap Key BLOB。其中 Public Key BLOB 用于传输公钥、Private Key BLOB 用于传输公私密钥对,Simple BLOB 用于传输会话密钥,Symmertic Wrap Key BLOB 用于导出或导入经过其他对称密钥加密后的对称密钥。

bType(密钥类型)	bVersion(版本号)	保留字	aiKeyAlg(算法 ID)

图 9-9 BLOBHEADER 的数据结构

以 Private Key BLOB 为例,它的数据结构包括了 BLOBHEADER 和私钥信息,如图 9-10 所示。私钥信息说明了 RSA 加密运算的各个参数,其中 Modulus 是 Prime1 和 Prime2 的乘积,若 Prime1 是 p,Prime2 是 q,则 Exponent1 为 d mod (p-1),Exponet2 为 d mod (q-1),coefficient 为 q 的逆 mod p,PrivateExponet 为 d。

BLOBHEADER	RSAPUBKEY	Modulus	Prime1	Prime2	Exponent1	Exponent2	Coefficient	PrivateExponet

图 9-10 Private Key BLOB 的数据结构

下面是一个 Private Key BLOB 的例子：

```
0x00000000 07 02 00 00 00 a4 00 00 52 53 41 32 00 02 00 00
0x00000010 01 00 01 00 6b df 51 ef db 6f 10 5c 32 bf 87 1c
0x00000020 d1 4c 24 7e e7 2a 14 10 6d eb 2c d5 8c 0b 95 7b
0x00000030 c7 5d c6 87 12 ea a9 cd 57 7d 3e cb e9 6a 46 d0
0x00000040 e1 ae 2f 86 d9 50 f9 98 71 dd 39 fc 0e 60 a9 d3
0x00000050 f2 38 bb 8d 5d 2c bc 1e c3 38 fe 00 5e ca cf cd
0x00000060 b4 13 89 16 d2 07 bc 9b e1 20 31 0b 81 28 17 0c
0x00000070 c7 73 94 ee 67 be 7b 78 4e c7 91 73 a8 34 5a 24
0x00000080 9d 92 0d e8 91 61 24 dc b5 eb df 71 66 dc e1 77
0x00000090 d4 78 14 98 79 44 b0 19 f6 f0 7d 63 cf 62 67 78
0x000000a0 d0 7b 10 ae 6b db 40 b3 b2 eb 2e 9f 31 34 2d cb
0x000000b0 bf a2 6a a6 1f e9 03 42 f2 63 9b b7 33 d0 fe 20
0x000000c0 83 26 1f 56 a8 24 f5 6d 19 51 a5 92 31 e4 2b bc
0x000000d0 11 c8 26 75 a0 51 e9 83 ca ee 4b f0 59 eb a4 81
0x000000e0 d6 1f 49 42 2b 75 89 a7 9f 84 7f 1f c3 8f 70 b6
0x000000f0 7e 06 5e 8b c9 53 65 80 b7 16 f2 5e 5e de 0b 57
0x00000100 47 43 86 85 8a fb 37 ac 66 34 ba 09 1a b1 21 0b
0x00000110 aa fa 6c b7 75 a7 3e 23 18 58 95 90 b5 29 a4 1e
0x00000120 15 76 52 56 bb 3d 6b 1d 2a d1 9f 5c 8a c0 55 ea
0x00000130 c3 29 a2 1e
```

以 Private Key BLOB 的数据结构对上述私钥进行解析，解析后的结构如表 9-28 所示。

表 9-28 对 Private Key BLOB 解析后的结果

参　　数	值
modulus	6b df 51 ef db 6f 10 5c 32 bf 87 1c d1 4c 24 7e e7 2a 14 10 6d eb 2c d5 8c 0b 95 7b c7 5d c6 87 12 ea a9 cd 57 7d 3e cb e9 6a 46 d0 e1 ae 2f 86 d9 50 f9 98 71 dd 39 fc 0e 60 a9 d3 f2 38 bb 8d
prime1	5d 2c bc 1e c3 38 fe 00 5e ca cf cd b4 13 89 16 d2 07 bc 9b e1 20 31 0b 81 28 17 0c c7 73 94 ee
prime2	67 be 7b 78 4e c7 91 73 a8 34 5a 24 9d 92 0d e8 91 61 24 dc b5 eb df 71 66 dc e1 77 d4 78 14 98
exponent1	79 44 b0 19 f6 f0 7d 63 cf 62 67 78 d0 7b 10 ae 6b db 40 b3 b2 eb 2e 9f 31 34 2d cb bf a2 6a a6
exponent2	1f e9 03 42 f2 63 9b b7 33 d0 fe 20 83 26 1f 56 a8 24 f5 6d 19 51 a5 92 31 e4 2b bc 11 c8 26 75
coefficient	a0 51 e9 83 ca ee 4b f0 59 eb a4 81 d6 1f 49 42 2b 75 89 a7 9f 84 7f 1f c3 8f 70 b6 7e 06 5e 8b
privateExponent	c9 53 65 80 b7 16 f2 5e 5e de 0b 57 47 43 86 85 8a fb 37 ac 66 34 ba 09 1a b1 21 0b aa fa 6c b7 75 a7 3e 23 18 58 95 90 b5 29 a4 1e 15 76 52 56 bb 3d 6b 1d 2a d1 9f 5c 8a c0 55 ea c3 29 a2 1e

下面以一个简单的数据加密应用，介绍使用 CryptoAPI 的流程。简单的来说，一个应用 CryptoAPI 加解密的应用程序可以分为如下步骤：首先获得 CSP 的句柄，创建密钥容器为存储密钥开辟空间；然后产生密钥，将密钥保存在密钥容器中，同时获得密钥句柄；设置密钥的各种参数，为加解密做准备工作；进行加解密操作，得到明文或密文。

图 9-11 说明了加密的实现过程，实现细节如下。

(1) 获得 CSP 句柄和创建密钥容器。应用程序首先指定 CSP 的名称、类型和密钥容器的名称，通过调用 CryptAquireContext()，获得 CSP 的句柄并创建密钥容器。

(2) 获得会话密钥句柄。应用程序通过调用 CryptGenKey()，获得随机产生的会话密钥，用于后续的加密过程，密钥以句柄的形式给出。

(3) 设置密钥参数。通过调用 CryptSetKeyParam()设置密钥的各种参数，如加密算法、初始化向量、填充模式等。

(4) 加密明文。用步骤 1 中获得 CSP 句柄和步骤 2 中获得的密钥句柄，通过调用 CryptEncrpyt()函数加密明文，得到密文。

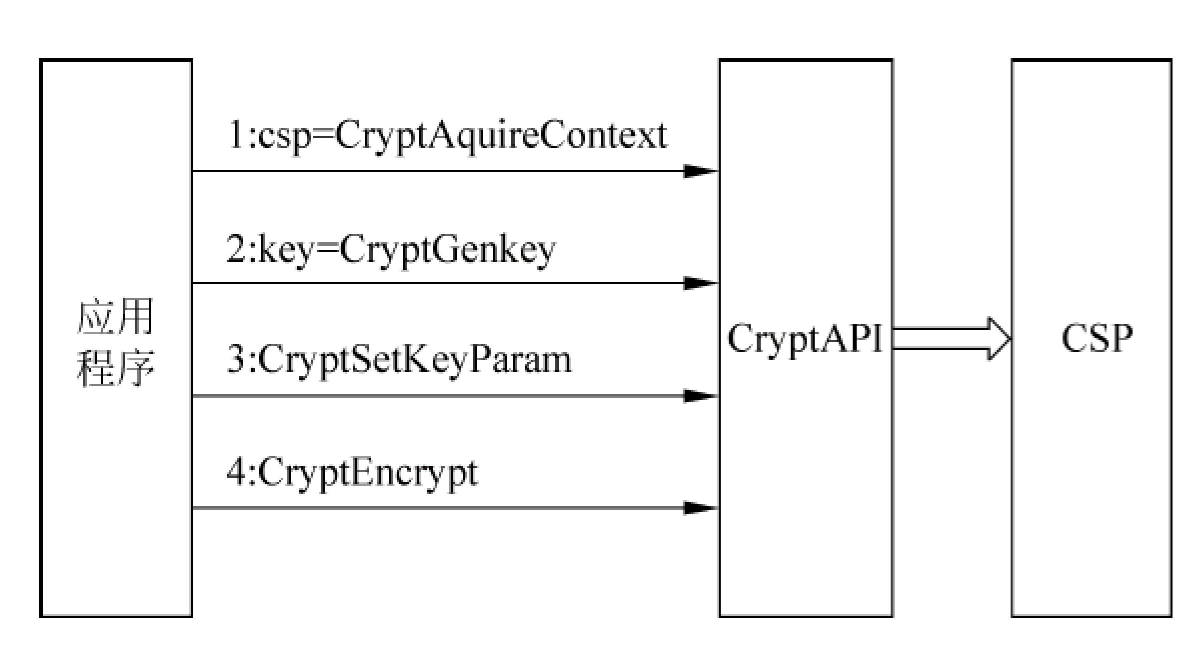


图 9-11 采用 CryptoAPI 加密的流程

实现的代码如下所示：

```
//获得 CSP 句柄和创建密钥容器
HCRYPTPROV hCryptProv = NULL;                              // 创建 CSP 句柄
LPCSTR UserName = "MyKeyContainer";                         // 定义密钥容器名称
//--------------------------------------------------------
//尝试获得 CSP 句柄和密钥容器.CSP 选用默认 CSP
//类型为 RSA_FULL,尝试打开一个已经存在的密钥容器
if(CryptAcquireContext(
    &hCryptProv,                                           // CSP 句柄
    UserName,                                              // 密钥容器名称
    NULL,                                                  // 使用默认 CSP
    PROV_RSA_FULL,                                         // CSP 类型
    0))                                                    // 标识值
{
     printf("A crypto context with the %s key container \n", UserName);
     printf("has been acquired.\n\n");
}
else
{
//--------------------------------------------------------
```

```
//如果获得 CSP 句柄失败,可能是因为该密钥容器不存在
//因此创建一个新的密钥容器
  if (GetLastError() == NTE_BAD_KEYSET)
  {
    if(CryptAcquireContext(
        &hCryptProv,
        UserName,
        NULL,
        PROV_RSA_FULL,
        CRYPT_NEWKEYSET))
      {
        printf("A new key container has been created.\n");
      }
      else
      {
        printf("Could not create a new key container.\n");
        exit(1);
      }
    else
    {
        printf("A cryptographic service handle could not be acquired.\n");
        exit(1);
    }
}                                                               // End of else.
//-----------------------------------------------
//创建会话密钥
BYTE * iv = NULL;
BYTE * pbData = NULL;
HCRYPTKEY hKey = NULL;
if(!CryptGenKey(hCryptProv, CALG_3DES, CRYPT_EXPORTABLE, &hKey)) {
      printf("Error %x during CryptGenKey!\n", GetLastError());
      goto done;
  }
//-----------------------------------------------
//设置会话密钥参数(若无需设置,则该步骤可省略)
//设置初始向量 IV = 0x1010101010101010
iv = new byte[8];
memset(iv,0x10,8);
if(!CryptSetKeyParam(hKey, KP_IV, iv, 0)) {
       printf("Error %x during CryptSetKeyParam!\n", GetLastError());
       goto done;
  }
//-----------------------------------------------
//加密明文
DWORD dwDataLen = 16;
DWORD dwBufLen = 16;
pbData = new BYTE[16];
memcpy(pbData,"Hello World!",13);

if(!CryptEncrypt(hKey,hHash,true,0,pbData,&dwDataLen,dwBufLen)){
      AfxMessageBox("Error %x during CryptEncrypt!\n", GetLastError());
```

```
        goto done;
    }
    done:
    //---------------------------------------------------
    // 释放密钥句柄和 CSP 句柄
    if(hKey != 0) CryptDestroyKey(hKey);
    if(hCryptProv != 0)  CryptReleaseContext(hCryptProv,0);
    if(iv != NULL)
        delete [] iv;
    if(pbData != NULL)
        delete []pbData;
```

9.3.2 数字信封应用

数字信封中消息的发送者采用对称加密算法加密明文,利用接收方的公钥加密密钥,将加密后的密钥与密文封装成一定的数据格式作为数字信封,发送给接收方。数字信封技术结合了对称加密和非对称加密两种方法的优点:由于待加密的明文数据量较大,采用对称加密方法可以显著地提高速度;由于对密钥的安全性要求较高,采用非对称加密方法可以增强密钥的安全性。数字信封为在不安全信道上传递秘密消息提供了一种可靠的手段。数字信封技术的实现过程如图 9-12 所示。

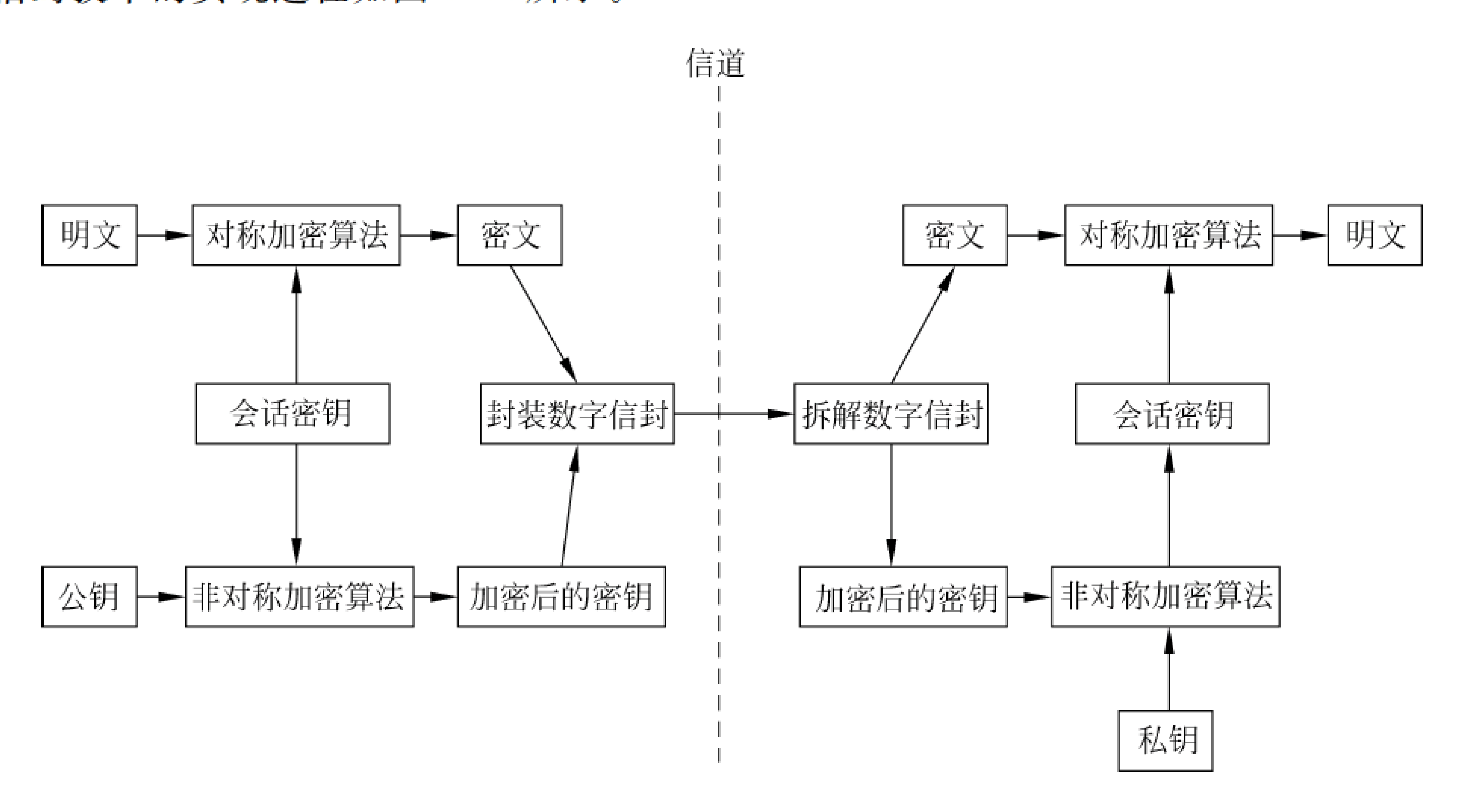


图 9-12 数字信封技术

开发数字信封应用软件首先需要定义数字信封文件的格式,约定数字信封的头 4 个字节用于保存加密后会话密钥的长度;接着是加密后的会话密钥;最后一部分为加密后的密文。数字信封的数据格式如图 9-13 所示。

密钥长度(4 字节)	加密后的会话密钥	密文

图 9-13 数字信封的格式

生成数字信封的流程简述如下:发送方首先获得 CSP 句柄,而后产生一个随机的会话密钥,然后用该会话密钥加密明文得到密文;再打开接收方预先给发送方的数字证书,从中

获得公钥，将 CSP 中存储的会话密钥导出，保存为 KEY BLOB 格式，用公钥加密会话密钥；最后依照图 9-13 所示的格式进行封装，生成数字信封。

解密数字信封的流程为：接收方收到数字信封后，首先拆解数字信封，依据头 4 个字节确定加密后的密钥长度，从数字信封中取出密钥；依据数字证书，获得私钥；获得 CSP 句柄，解密会话密钥，并将解密后的会话密钥导入 CSP；利用会话密钥解密密文，得到明文。

采用 CryptoAPI，3DES 和 RSA1024 加密方法实现生成数字信封的详细过程如下。

(1) 从输入界面获得明文。

(2) 调用 CryptAcquireContext()获得指定的 CSP 句柄，并创建密钥容器。

(3) 调用 CryptGenKey()生成随机会话密钥，加密算法指定为 CALG_3DES。

(4) 调用 CryptEncrypt()加密明文，获得密文，并将密文保存到临时文件或内存中。

(5) 从输入界面获得用户选择的证书，调用 CertOpenSystemStore()打开指定的证书库，调用 CertEnumCertificatesInStore()获得证书上下文。

(6) 调用 CryptImportPublicKeyInfo()获得证书中的公钥。

(7) 调用 CryptExportKey()将会话密钥从 CSP 中导出，并用步骤 6 中获得的公钥进行加密，加密后的结果以 KEY BLOB 的格式保存。

(8) 获得 KEY BLOB 的字节长度，将字节长度、KEY BLOB 和步骤 4 中获得密文，依照图 9-13 的数据格式进行封装，生成数字信封。

拆解数字信封获得明文的过程如下。

(1) 从输入界面获得数字信封数据，拆解数字信封。

(2) 从输入界面获得用户选择的数字证书，调用 CertOpenSystemStore()打开指定证书，调用 CertEnumCertificatesInStore()获得证书上下文。

(3) 调用 CryptAcquireCertificatePrivateKey()获得证书私钥，并同时获得 CSP 句柄。

(4) 调用 CryptImportKey()，将步骤 1 中获得的加密会话密钥解密，并将其导入 CSP。

(5) 调用 CryptDecrypt()用解密后的会话密钥解密密文，得到明文。

我们采用 C++语言在 VISUAL STUDIO 2005 的环境下了开发了一套数字信封应用演示系统，其界面如图 9-14 所示。例如要将明文"智能卡安全与设计"封装为数字信封，其流程为：将明文输入到编辑框中，在菜单栏中选择【数字信封】|【加密】，在证书选择对话框中选择相应证书，界面如图 9-15 所示。封装后得到的数字信封则在界面中显示，如图 9-16 所示。

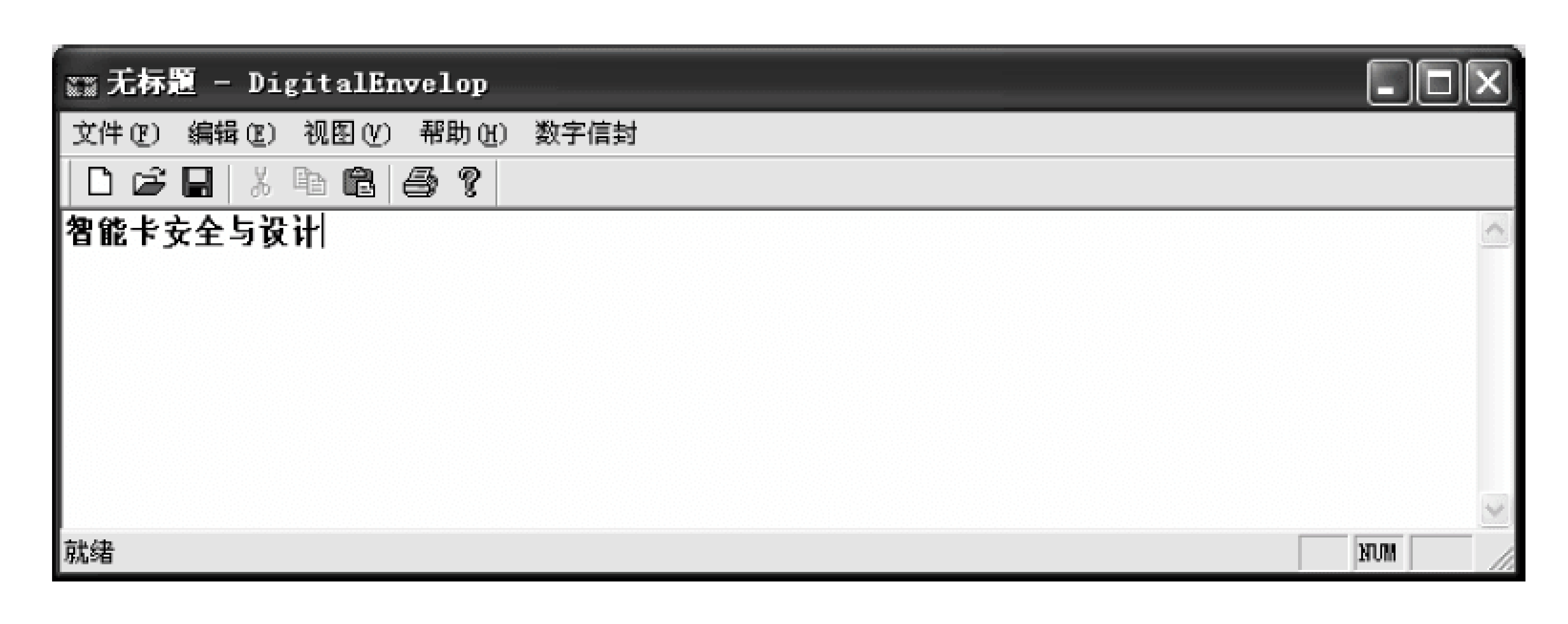


图 9-14 在数字信封中输入明文

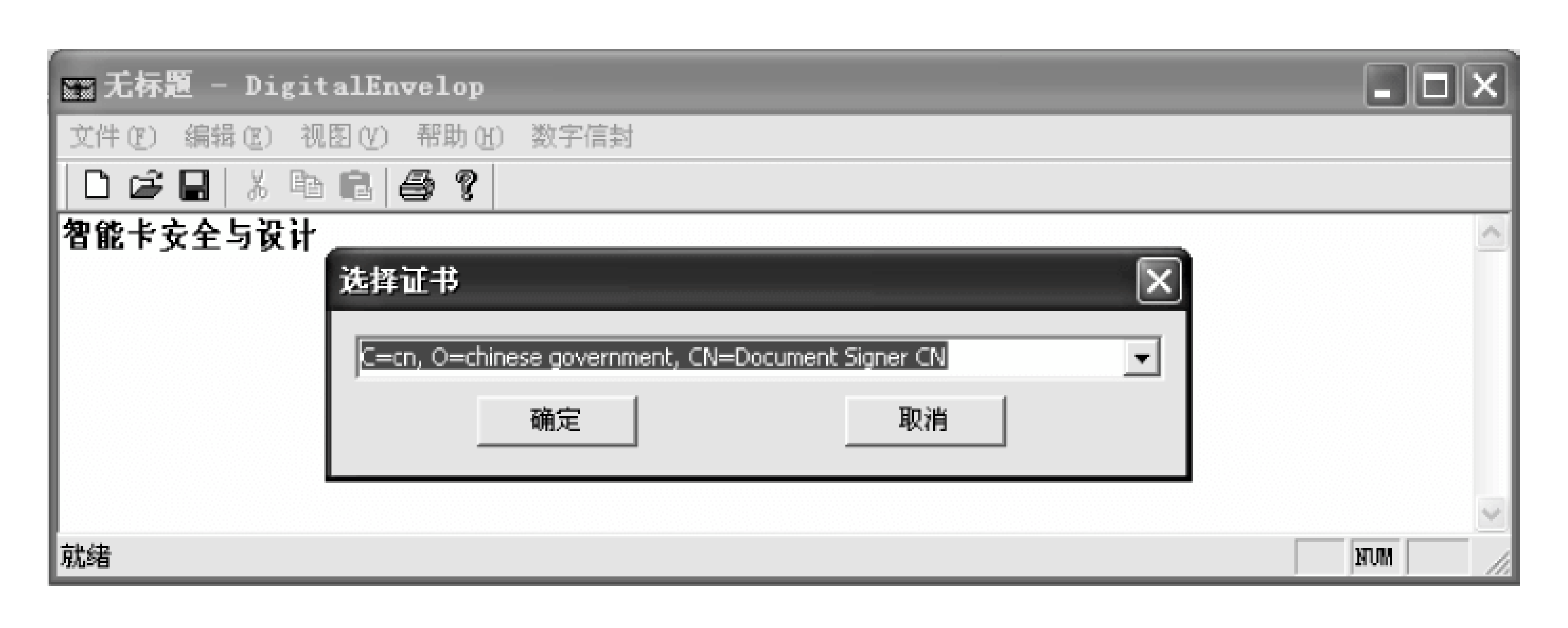


图 9-15 选择数字证书



图 9-16 生成数字信封

接收方收到数字信封时，首先将数字信封数据输入到界面中，在菜单栏中选择【数字信封】|【解密】，在证书选择对话框中选择相应证书，则可得到解密后的明文。

9.3.3 SOD 生成与验证

证件安全对象(document secure object，SOD)是在电子证件中广泛应用的一种文件形式，其主要作用是防止对证件中的数据进行篡改。SOD 中存储着电子证件其他基本文件(elementary file，EF)的 Hash 值和，以及经文件签名者(document signer，DS)签名的结果，SOD 中同时还保存着 DS 的数字证书，用于对 DS 签名值进行认证。SOD 文件的格式如图 9-17 所示。



图 9-17 SOD 文件的格式

SOD 与被动认证(passive authenticate，PA)是紧密结合在一起的，通过 PA 认证可以防止攻击者篡改 EF 中的数据，当 EF 中的数据被篡改后，其 Hash 值将发生变化，与 SOD 中存储的 Hash 值不同，便可检验 EF 是否被篡改。由于用户无法获得 DS 证书的私钥，因此无法通过修改 Hash 值而后重新签名的方法来伪造 SOD。

SOD 生成的步骤为：首先计算各 EF 的 Hash 值，得到未签名的 SOD；然后用 DS 数字证书的私钥对未签名的 SOD 进行签名，将 DS 公钥数字证书、计算后得到的数字签名以及未签名的 SOD 以图 9-17 所示的数据结构进行封装，得到签名后的 SOD。签名后的 SOD 可以作为一个文件存储在智能卡中，当需要进行 PA 认证时，从智能卡中读出即可。

SOD 验证的步骤为：首先从智能卡中读取出 SOD 数据，用 SOD 中存储的数字证书的

公钥对 SOD 中的数字签名进行签名验证,验证结果作为 SOD 验证的结果。

采用 CryptoAPI 方法实现 SOD 生成的详细步骤如下。

(1) 获得各 EF 的数据,调用 CryptCreateHash()函数计算各 EF 的 Hash 值,算法标识设置为 CALG_SHA1。

(2) 依照未签名 SOD 格式将计算得到的各 EF 的 Hash 值进行封装。

(3) 从输入界面获得用户选择的数字证书,调用 CertOpenSystemStore()打开指定证书,调用 CertEnumCertificatesInStore()获得证书上下文。

(4) 调用 CryptSignMessage()函数两次,计算数字签名(第一次调用计算签名后的数据长度,分配内存;第二次调用计算签名结果)。同时封装为如图 9-17 所示的 SOD 文件格式。

SOD 验证的步骤如下。

(1) 获得 SOD 数据。

(2) 调用 CryptVerifyMessageSignature()两次(作用类同 SOD 生成步骤 4),对签名进行验证,得到签名认证结果。

采用 C++语言,在 VISUAL STUDIO 2005 环境中开发的 SOD 生成与验证软件,界面如图 9-18 所示。

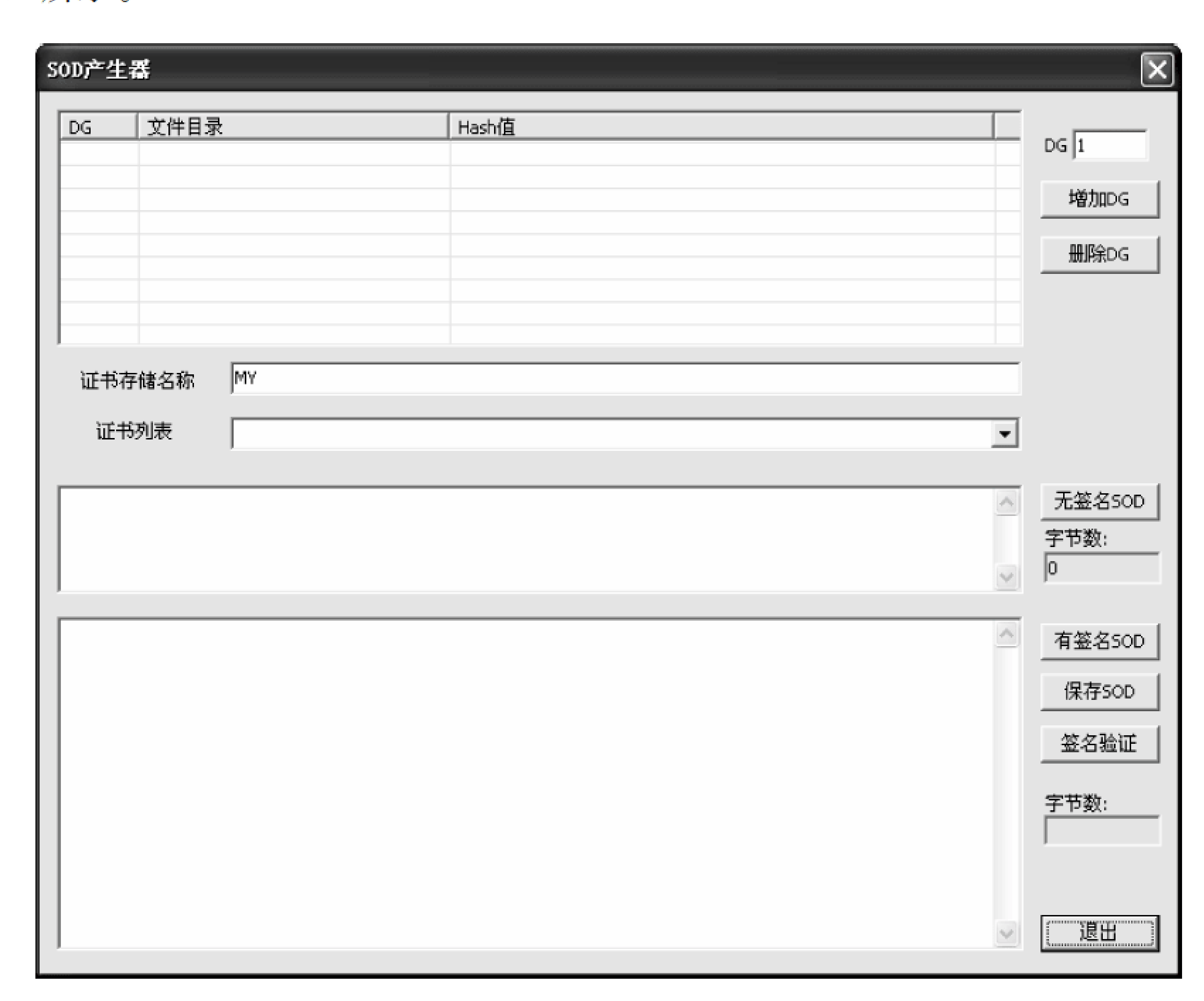


图 9-18 SOD 生成与验证软件界面

其中【增加 DG】按钮用于获取 EF 数据,并计算其 Hash 值。【无签名 SOD】用于将计算后得到的各 EF 的 Hash 值封装成无签名 SOD 的格式。【有签名 SOD】根据用户选择的证书对无签名 SOD 进行签名,最后将无签名 SOD、数字证书和数字签名封装为 SOD 的格式,并在界面显示。【签名验证】用于对 SOD 进行签名验证。

9.4 CSP 开发

9.4.1 CSP 开发简介

由于受到密钥长度、加密算法等安全因素的影响，往往需要自行开发 CSP。CSP 的实现可以采用软件或硬件的形式，也可以将两种方式结合起来，软件实现部分 CSP 功能，其他部分由硬件实现。如果开发在 Windows 操作系统上应用的 CSP，则需要实现 CryptoSPI 中规定的接口函数，也可以为应用程序提供专用的二次开发接口，如图 9-19 所示。

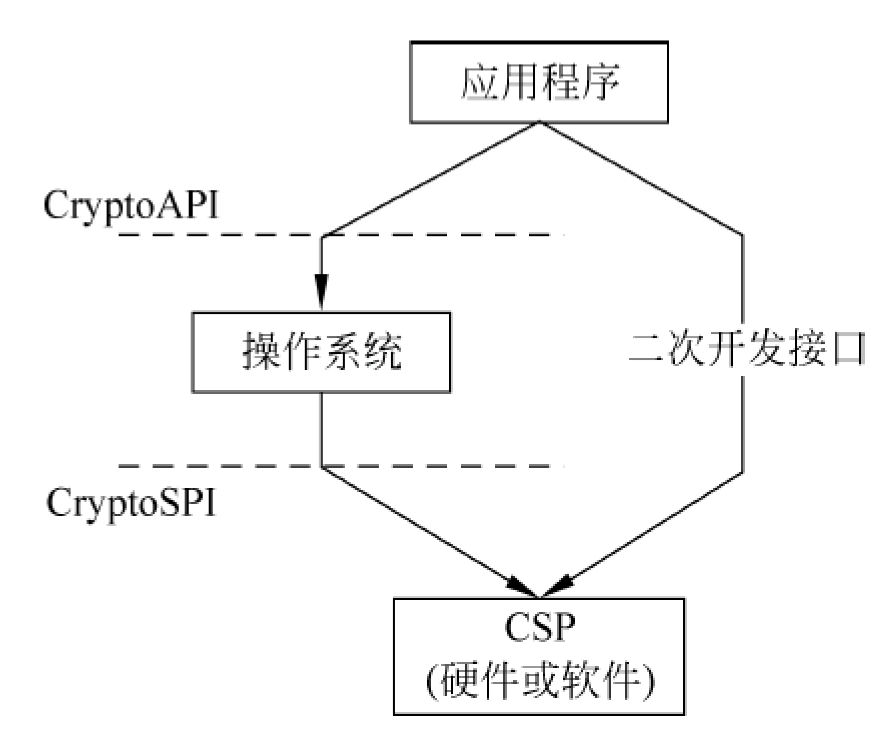


图 9-19 调用 CSP 的两种方式

软件实现采用编程语言开发软件代码，实现 CSP 功能。为了加快开发进度，可以采用已经成熟的加密库，如 OpenSSL 等。OpenSSL 提供了加解密算法和 SSL（安全套接字层）服务的源代码和丰富的接口。CSP 可以通过调用 OpenSSL 中的各种函数，实现加解密算法、Hash 算法、公钥和会话密钥算法、签名等算法。

硬件实现可采用加密机、智能卡、USBKey 等形式。加密机数据处理能力强，速度快，但是其体积大，价格昂贵，主要应用在数据量很大，不需要有便携性的服务器领域。智能卡由于需要专用读写机具的配合，因此降低了其通用性。USBKey 采用 USB 接口直接与计算机通信，是一种常见的 CSP 实现形式。

图 9-20 所示为 USBKey 中的文件结构，一个 USBKey 的文件系统只有一个 MF，MF 下可创建一个或多个 DF，每个 DF 对应一个 CSP，同时 MF 下创建了一个 EF（manager key），用于保存管理者密钥，以及获得对各 CSP 进行操作的权限。各 CSP 下可创建一个或多个 DF，作为密钥容器来保存密钥，同时在各 CSP 下创建一个目录型的 EF 文件 DIR，用于记录各 CSP 下的密钥容器的文件名。各密钥容器文件下保存有一个或多个 EF，用于保存密钥。

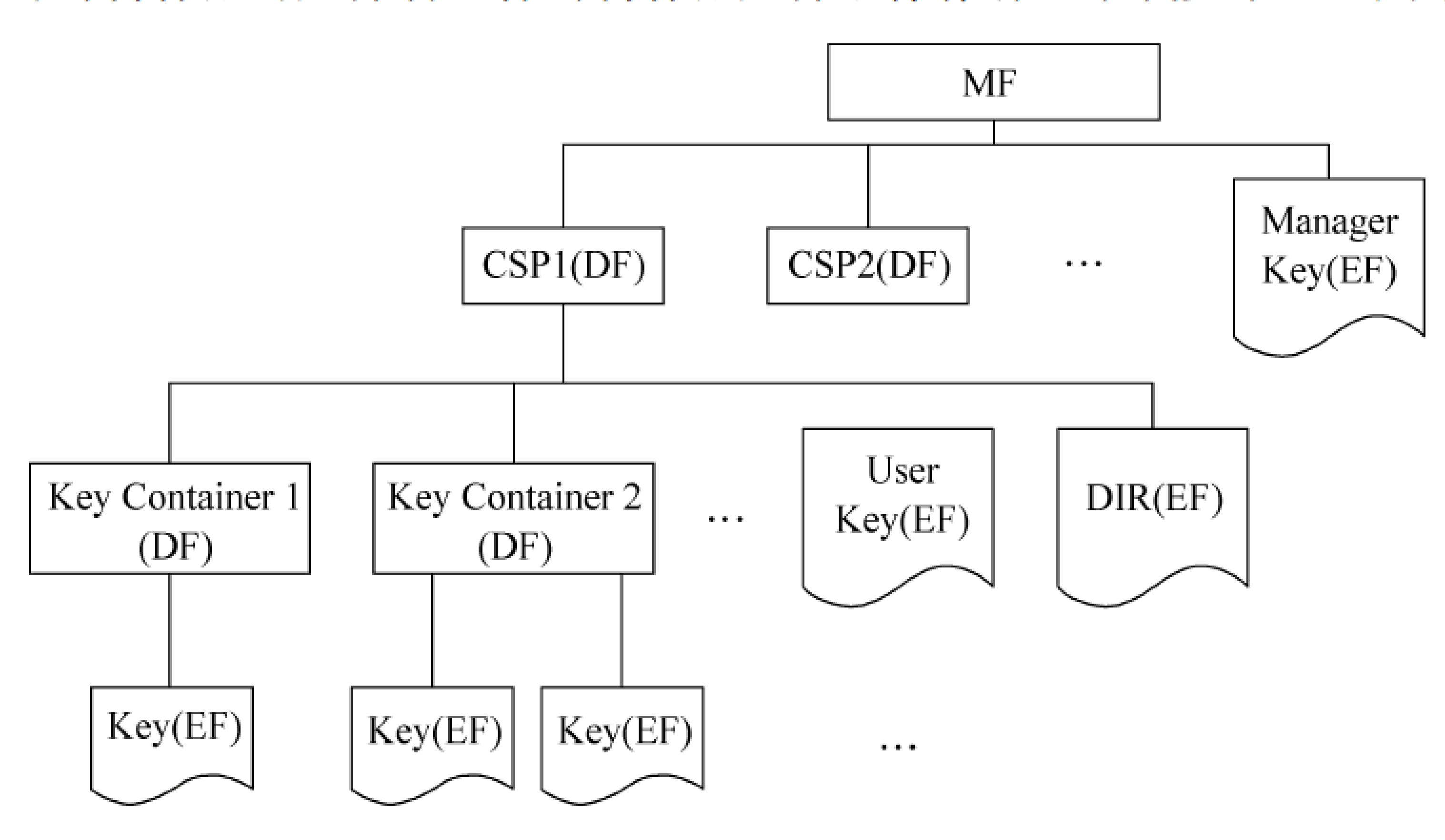


图 9-20 USBKey 中的文件结构

软硬件结合可将 CSP 的部分功能由软件实现,其余部分由硬件实现。例如非对称加密算法速度较慢,可采用硬件的形式实现,提高其速度;对称算法加密速度较快,可采用软件的形式实现。

9.4.2 CryptoSPI 接口函数

开发在 Windows 操作系统上应用的 CSP 时,开发人员要实现 CryptoSPI 中规定的 25 个接口函数,可以将其分为 4 部分,如表 9-29~表 9-32 所示。

1. CSP 连接函数

提供了一组函数,用于与 CSP 连接和断开连接。函数描述如表 9-29 所示。

表 9-29 CSP 中的连接 API

函数名称	功能描述
CPAcquireContext	获取指定 CSP 的密钥容器的句柄
CPGetProvParam	恢复 CSP 的属性
CPReleaseContext	释放由 CPAcquireContext 获得的句柄
CPSetProvParam	设置 CSP 的属性

2. CSP 密钥产生和交换函数

提供了一组函数,用于产生密钥,与其他用户交换密钥,产生、配置和销毁密钥。函数描述如表 9-30 所示。

表 9-30 密钥产生和交换 API

函数名称	功能描述
CPDeriveKey	从一个密码派生一个密钥
CPDestroyKey	销毁一个密钥
CPDuplicateKey	复制密钥,包括密钥状态
CPExportKey	将密钥以 Key BLOB 格式导入到应用程序的内存中
CPGenKey	创建一个随机密钥
CPGenRandom	创建一个随机数
CPGetKeyParam	恢复密钥参数
CPGetUserKey	获得交换密钥或签名密钥的句柄
CPImportKey	将以 Key BLOB 格式存在的密钥导入到 CSP 中
CPSetKeyParam	设置密钥参数

3. CSP 数据加密函数

提供了一组函数,用于执行加密和解密操作。函数描述如表 9-31 所示。

表 9-31 数据加解密 API

函数名称	功能描述
CPDecrypt	用指定的密钥解密文件
CPEncrypt	用指定的密钥加密文件

4. CSP 哈希和数字签名函数

提供了一组函数,用于计算 Hash 值和产生与验证数字签名。函数描述如表 9-32 所示。

表 9-32 Hash 运算和签名 API

函数名称	功能描述
CPCreateHash	创建一个 Hash 对象，并返回它的句柄
CPDestroyHash	销毁 Hash 对象
CPDuplicateHash	复制一个 Hash 对象，包括 Hash 对象的状态
CPGetHashParam	恢复 Hash 对象的参数
CPHashData	对一块数据进行 Hash 运算，并把它添加到指定的 Hash 对象中
CPHashSessionKey	Hash 一个会话密钥，并把它添加到指定的 Hash 对象中
CPSetHashParam	设置 Hash 对象的参数
CPSignHash	签名指定的 Hash 对象
CPVerifySignature	验证签名

另外还需要实现两个 CSP 的注册接口。用它们来完成向注册表写入 CSP 信息的功能，即 CSP 的安装。CSP 可以通过两种方式来安装：一种是编写单独的安装程序，另一种是在 DLL 文件中增加 DllRegisterServer()和 DllUnregisterServer()两个接口，通过 resgsvr32 来完成 CSP 的安装和卸载。

9.4.3 CSP 开发流程

通用 CSP 的开发流程：

(1) 从微软网站上下载 CSP 开发工具包 CSPDK。

(2) 创建 CSP. dll，导出 CrytoSPI 函数接口。

(3) 开发 CSP 安装程序，创建合适注册表项。

(4) 采用软件、硬件或软硬件结合的方式实现 CryptoSPI 函数的功能。

(5) 测试所开发的 CSP. dll 的实现功能。

(6) 用 CryptoAPI 测试开发出的 CSP。

(7) 得到微软对该 CSP 的正式签名。

(8) 测试经过微软正式签名 CSP。这个步骤和(4)相同，不过此时的 CSP 已经通过了微软的正式签名。

参考文献

[1] 王育民，刘建伟. 通信网的安全：理论与技术. 西安：西安电子科技大学出版社，1999

[2] 王爱英. 智能卡技术. 第 2 版. 北京：清华大学出版社，2000

[3] Bruce E 著，刘宗田等译. C++编程思想第一卷：标准 C++导引. 第 2 版. 北京：机械工业出版社，2000

[4] Douglas R S，冯登国译. 密码学原理与实践. 第 2 版. 北京：电子工业出版社，2003

[5] Tom S D，Simon J，沈晓斌译. 程序员密码学. 北京：机械工业出版社，2007

[6] 康荣保，罗慧，何志平等. 利用 CryptoAPI 对硬件 CSP 操作的分析. 信息安全与通信保密，2005，(11)：65～68

[7] 李奇富. Microsoft CryptoAPI 安全框架分析及本地化策略. 微计算机信息，2006，22(9)：42～44

［8］ 赵彦峰.基于微软 CryptoAPI 的数字签名和数字信封的实现.信息安全与通信保密，2007，(12)：83～85
［9］ 裴照君，段哲民，王海涛.基于 Microsoft CryptoAPI 框架下的数据安全通信系统开发.微型电脑应用，2008，24(10)：21～24
［10］ 常青，张卡，张其善.基于智能卡的 CSP 的设计与实现.计算机工程与应用，2005，41(5)：117～119
［11］ 冉春玉，汪学舜，吕恢艳.加密服务提供(CSP)的实现与开发.武汉理工大学学报，2003，25(10)：87～89
［12］ 陈由甲，徐家恺.USBKey 软硬件结合的 CSP 设计与实现.科学技术与工程，2007，7(16)：4055～4057
［13］ 宋玲，李陶深，陈拓.在 VC++中用 CryptoAPI 保证安全数据通信.计算机应用与软件，2005，22(7)：29～30



应用系统设计

智能卡在现代人类社会拥有极其广阔的应用市场和发展前景，如城市公共交通、智能通信、物流管理、身份认证、门禁考勤、小额消费等等，几乎延伸涉及人类生活的各个领域。为帮助读者对智能卡应用有更加深入直观的了解，下面具体介绍多种国内常见智能卡的典型应用，并给出相应的设计思路与方案。

10.1 市政公交一卡通

城市公共交通一卡通系统是一项信息系统工程项目，是实现信息化交通管理的基础。一卡通系统能够解决长期困惑公共交通行业和部分非公共交通行业的自动收费问题，最终可以实现“一卡在手，走遍全城”的梦想。

随着信息技术和IC卡技术的发展，IC卡在公共交通上的应用越来越广泛。国内外的很多城市已经实现了市政公交一卡通的实际应用。1997年，香港八达通卡系统投入使用。在该系统中，乘客使用非接触式IC卡能够搭乘各种公共交通运输工具，包括地铁、高架铁路、轻型铁路、巴士、轮渡，也可用于交通运输以外的用途，例如停车、购物和通信等。1998年9月，上海“一卡通”工程正式启动，1999年5月，以交通卡清算处理中心和交通卡发卡充值中心为基础成立了上海东方交通卡股份有限公司，实现了统一发卡、统一融资和统一标准。到2007年上海已在市区超过2000条公交线路，地铁1号线、2号线全线（自动换乘），8条轮渡线路，超过3万辆出租车和明珠线轻轨上实现了公共交通卡的应用。

现在市政公交一卡通不仅能够完成公交、地铁、渡轮、出租等交通工具上的自动支付，还可以在ATM机上进行自动转账充值。随着CPU卡的不断普及，以及IC卡的安全性能不断提高，一些能够完成小额支付和相关电子商务功能的一卡通被越来越多的人所使用。近两年，“多市政公交一卡通”或者称为“区域一卡通”成为公交IC卡应用的又一热点议题，如2004年推出的“长沙易达通”系统的应用目标就是不仅囊括长沙市的公交、出租车等交通工具的付费，还能够完成各大中院校的课堂点名、保安门禁、图书证、校园费用支付，具备了电子钱包功能。以上海、南京、杭州为顶点的长三角地区和以北京、天津为中心的环渤海地区都将“区域一卡通”作为城市发展目标。

10.1.1 一卡通业务流程分析

现阶段市政公交一卡通的主要应用领域是公共交通领域，但也正在逐步扩展到与居民生活息息相关的其他领域。针对多领域多业务的需求，就要求一卡通能够有很好的兼容性、安全性和方便性。图10-1给出了一卡通的业务流程图。其中实线表示信息流向，虚线表示资金流向。

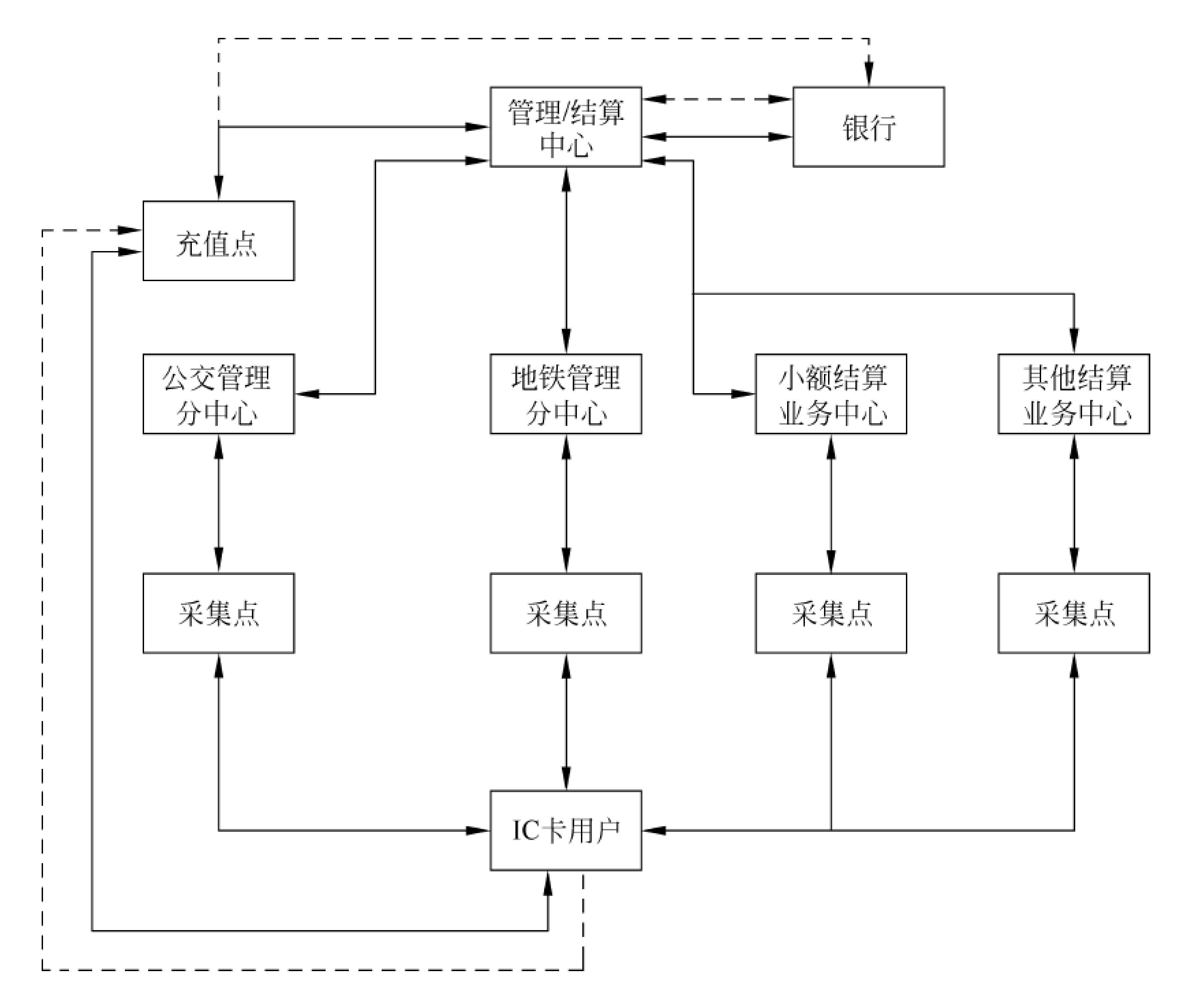


图 10-1 一卡通业务流程图

首先,管理/结算中心是管理所有涉及公共交通服务工作的 IC 卡应用管理总中心,它是一个负责市政公交一卡通应用系统的统一指挥、统一管理、统一结算、统一标准、统一维护和统一监控的高层次应用信息化管理组织。该中心负责所有 IC 卡的发行管理和系统密钥安全体系的管理和维护,能够有效地对市政公交一卡通在不同领域进行统一管理和授权,使得不同的应用领域在 IC 卡应用条件成熟时,能够加入这一系统并遵从系统的总体结构,最终达到通过科技手段公平、公正、廉洁自律地主持运营的目的。

其次公交管理分中心、地铁管理分中心和其他管理分中心(如小额结算业务中心,其他结算业务中心等)作为一卡通在各领域应用业务的管理分中心来为居民提供相应的便捷服务,负责本系统内部各核算单元的结算,完成付费操作,可以从管理/结算中心系统下载各种操作数据,将必要的数据传输到各个数据采集点。

数据采集点指的是在各行业中,为方便居民挂失、充值、消费等 IC 卡操作的管理点。以公交系统为例,各个公交总站设有一卡通的管理和充值点(即可被认为是数据采集点)。居民在数据采集点进行充值后,充值数据会汇总到管理/结算中心,通过管理/结算中心的调整,最终把数据传到银行数据系统中,完成了一卡通账户的充值过程。

10.1.2 一卡通系统构成

1. 网络系统设计

市政公交一卡通应用领域广泛,管理系统庞大,使用用户众多。以北京市为例,公交领域发卡累计已超过 3200 万张,每天 1000 万人次使用,需要进行数据传输的站点 800 个,每天产生的数据量巨大。因此,必须建设一个可靠性高、传输能力强、操作性好的网络系统,方

便整个网络的管理。

整体网络系统为三层拓扑结构，第一层为管理/结算中心的局域网，第二层为业务管理分中心，第三层为授权数据采集点。管理/结算中心与各个采集点间采用广域网连接，采集数据传送路径为：各站采集器→采集点路由器→DDN/ADSL 线路→分中心路由器→DDN/ADSL 线路→管理/结算中心路由器→结算主机。网络结构图如图 10-2 所示。

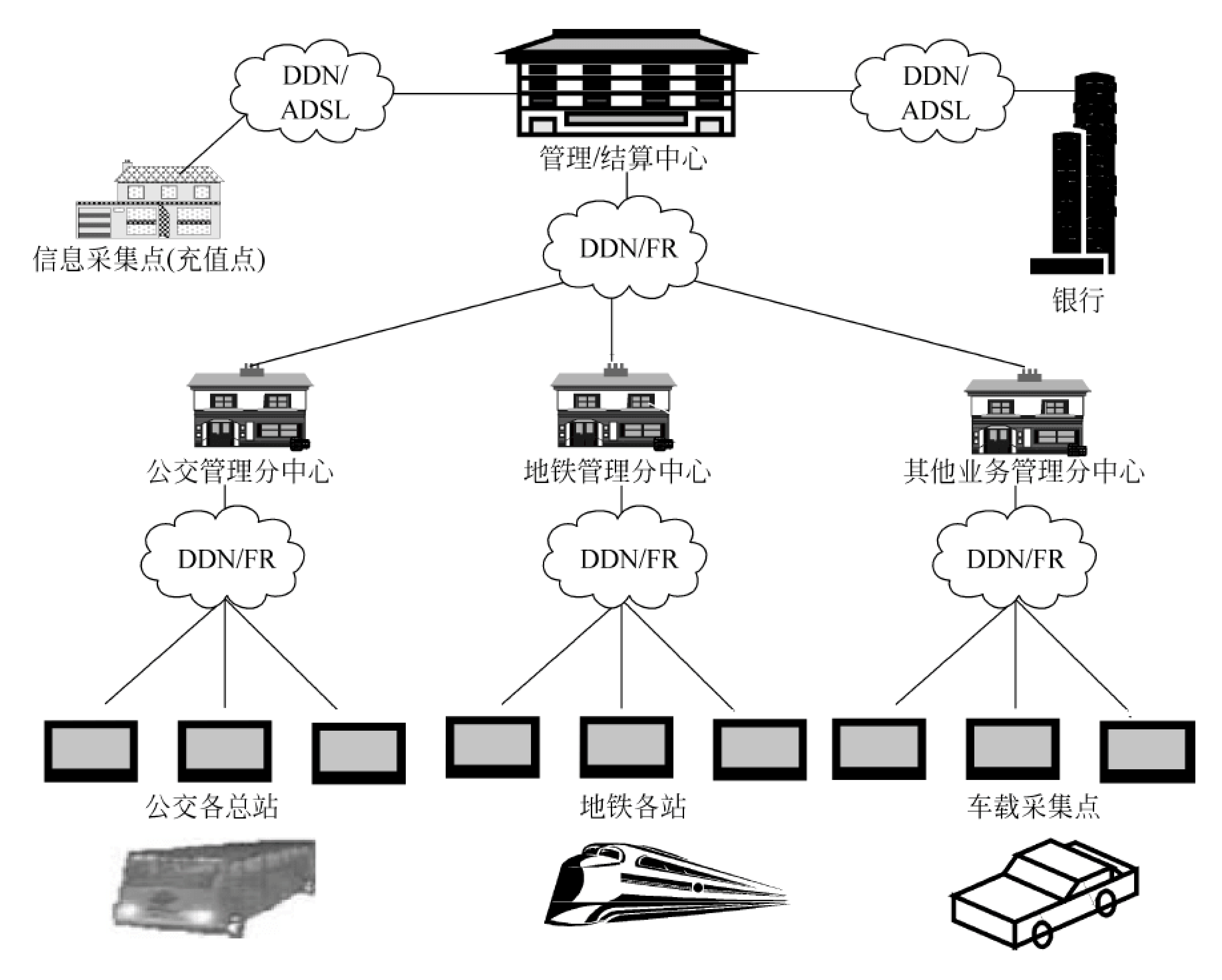


图 10-2　一卡通系统网络结构图

2. 应用系统设计

应用系统根据应用场合和任务性质的不同，整个系统可分为以下几部分。

整个市政公交一卡通的应用软件系统应该包含数据采集功能、交易处理功能、统计分析功能、结算功能、财务功能、综合管理功能等几大模块，分属于中心综合业务管理系统、分中心综合业务管理系统、中心结算系统、业主结算系统、本地数据处理系统等几个系统。它们之间的层次关系可参考图 10-3。

结算系统(包括中心结算系统和业主结算系统)是整个市政公交一卡通应用系统的核心，它的工作正常与否直接关系到系统是否成功。根据市政公交一卡通整体业务流程，结算系统又可划分为票卡管理、充值业务处理、消费业务处理、综合决策支持、结算和账务处理、乘客服务、运行管理、设备管理、设备运行监控、办公自动化、账务预处理、财务系统等 12 个大模块。

综合业务管理系统(包括中心综合业务管理系统和分中心业务管理系统)采用先进的客户/服务器结构和三层 B/S 结构。其中大多数业务模块采用客户/服务器结构；对消费处理、决策支持和乘客服务模块，利用中间件技术和数据仓库技术，提供三层 B/S 结构，以保

证应用系统的扩充和易管理。业务管理数据库和 Web 服务器分别采用独立的服务器。这种方式的优点在于结算数据同业务管理数据物理上是相互独立的，处理效率高，不会因为结算用服务器硬件的故障造成系统停止运行。同时，采用独立的 Web 服务器提供查询服务对于系统的安全提供了更可靠的保障。而这种方式的缺点是网络建设费用高，要解决相互之间的数据复制问题。在交易数据量不大且业务管理系统不繁忙的时候，采用这种方式会造成投资浪费。

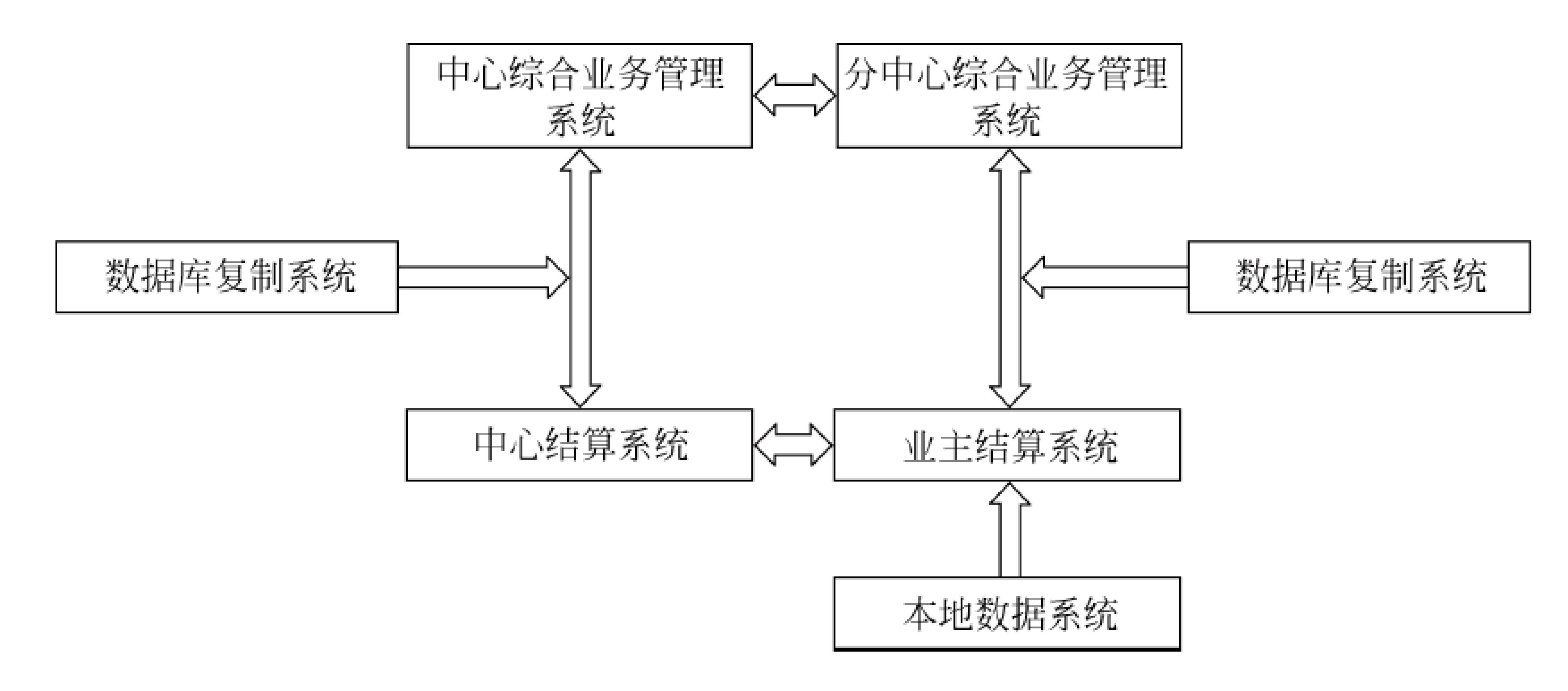


图 10-3 应用软件系统图

10.1.3 安全方案设计

市政公交一卡通系统涉及管理中心、合作银行、运营商、售卡点、充值点、数据采集点、持卡用户、司机等多个业务实体；业务处理涉及售卡、充值、脱机消费、资金结算等多个过程。由于一卡通的应用范围广，涉及点多，系统功能复杂，相应地对系统的安全可靠性要求也很高。

一卡通存在的安全隐患有：

(1) IC 卡中的密钥可能被破解，导致卡被伪造或非法使用。

(2) POS 终端可能被伪造，非法读取或修改用户 IC 卡上的信息。

(3) 因为用户消费交易过程是脱机的，攻击者可能使用时间差进行作案。例如：使用一张已经挂失的用户卡。

(4) 交易过程可能被截获、篡改，导致非法交易。

(5) 不同实体之间传输的数据可能被截获、篡改，导致非法数据，以及用户私有信息的泄露。

(6) 结算中心存放大量敏感信息，一旦结算中心网络或数据库主机被攻破，可能会造成系统数据泄露和整个应用系统的瘫痪。

针对以上安全隐患通常采用下述安全策略加以应对：

(1) 卡的密钥必须是一卡一密、一应用一密，以防止整个应用系统的安全体系被攻破，用户 IC 卡有唯一的卡序列号，在制卡过程中，主密钥根据该序列号分散得到用户 IC 卡上的子密钥，实现一卡一密的机制。这样即使一张用户卡的密钥被破解，不会对其他用户卡造成影响。

(2) 保证卡与读写机具之间的通信数据正确传输，防止通信数据被非法窃取或篡改。

(3) 采取 IC 卡与机具间的相互认证机制。

(4) 在密钥的管理方面采用如图 10-4 所示的密钥管理机制。

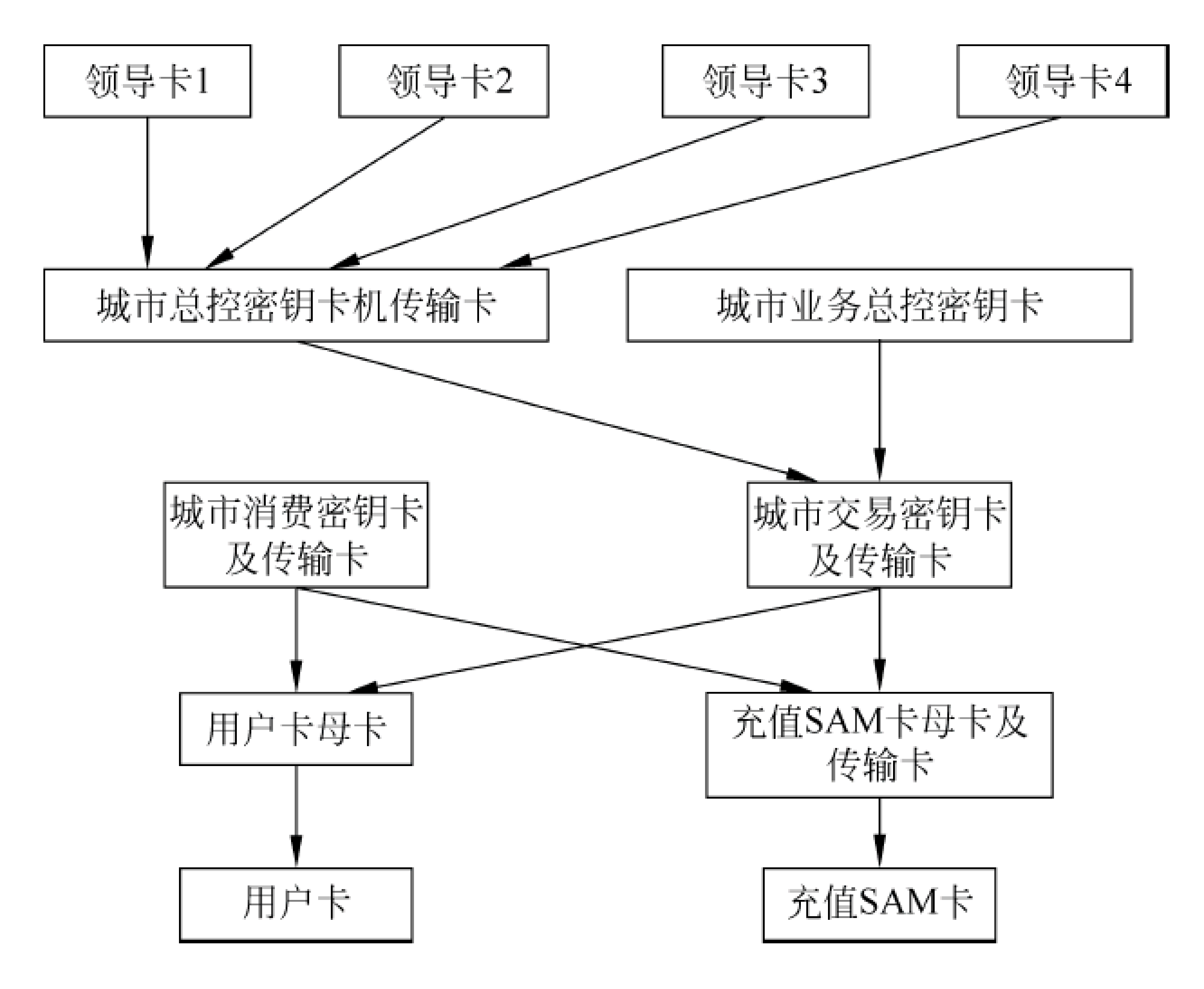


图 10-4 **密钥管理机制图**

随着城市的发展,一卡通将会在各个公用行业广泛运用,为建设数字化新城市做出有力的推动,为城市居民的生活提供更多的便利。

10.2 基于 RFID 标签的物流应用

射频识别(radio frequency identification,RFID)技术源于军事中的应用。在 20 世纪 70 年代主要应用于固定位置的物品的识别,来确定基础设施的归属。自 1998 年,麻省理工 Auto-ID 研究所开始研究跟踪与识别移动中的物品的新方法。他们在降低 RFID 标签的制造成本,为存储和传输大量数据优化数据网络,发展公开的标准等方面取得了重要的突破。

10.2.1 RFID 标签

RFID 标签分为有源和无源两种。有源 RFID 标签内部装有电池,体积较大,使用寿命相对较短,但因其内部的电源使得输出的电磁波能量大,传播距离远。无源 RFID 标签内部没有电池,通过阅读器发出的电磁波来为 RFID 标签提供能量,而带来的相应问题就是信号响应距离短,但其使用寿命长,无需更换电池,所以体积也可以做到很小,便于使用。有源 RFID 标签多用于贵重物品的物流领域,而对于普通货物来说,无源 RFID 标签的使用更加普遍。

RFID 标签比普通的 IC 卡存储空间要小,一般在几十到几千个字节不等,所存储的信息量较小,一般存储一个唯一编码,该编码通常作为该物品的唯一标识。由于只存储编码,所以在数据的读取过程中信息的传输简单方便,节省时间。

最基本的 RFID 标签系统由标签、阅读器和天线三部分组成。标签由耦合元件及芯片

组成。物品的标识存在于芯片的存储区域中。阅读器是读取标签信息的设备,可设计为手持式或固定式。天线用于在标签和阅读器件之间传递信号。这三部分的工作原理如图 10-5 所示。首先阅读器通过天线发送出一定频率的射频信号,当 RFID 标签进入阅读器的工作场时,其天线产生感应电流,从而 RFID 标签获得能量,激活后向阅读器发送出自身编码等信息。阅读器接收到来自标签的载波信号,对接收到的信号进行解调和解码,最后送入计算机进行处理。

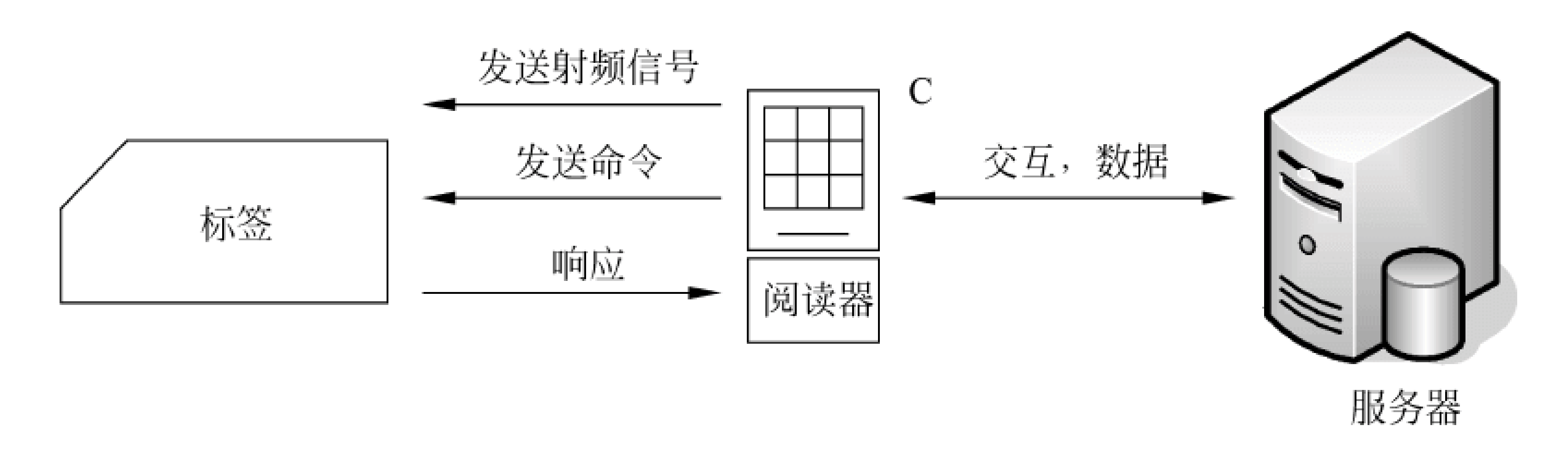


图 10-5 RFID 标签工作原理图

10.2.2 RFID 标签物流系统

1. 系统组成

基于 RFID 技术的分销物流管理系统架构主要由 4 部分组成:RFID 数据采集层、RFID 数据处理层、信息服务层以及分销物流管理应用层,如图 10-6 所示。

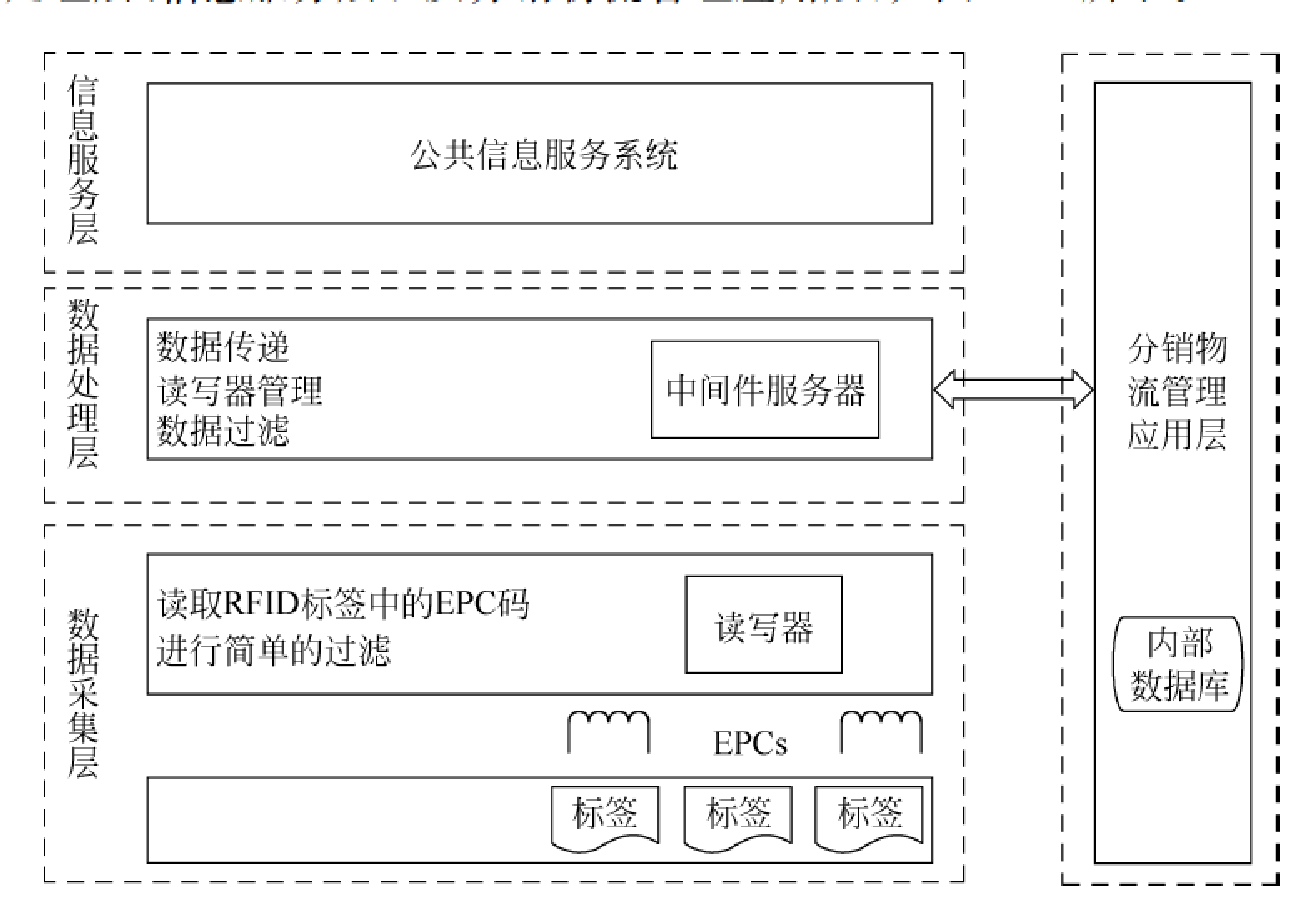


图 10-6 物流管理系统架构图

(1) RFID 数据采集层

RFID 数据采集层是基于 RFID 的分销物流管理系统的底层,它主要由 RFID 标签和读写器组成,读写器将采集到的 RFID 标签信息传递给中间件进行处理。

(2) RFID 数据处理层

RFID 数据处理层也被称为中间件服务器,是一个软件系统,扮演电子标签和应用程序

之间的中介角色，主要任务是对读写器读出的 EPC 进行传送和管理。它利用了一个分布式的结构，层次化地进行组织和管理数据流。每个层次上的中间件系统将收集、存储和处理信息，并与其他的中间件系统进行交流。实际应用时，在产品从成品仓库到专卖店再到客户的过程中，读写器将不断收到一连串的 EPC 码，由中间件对这些数据进行传送和管理。

(3) 信息服务层

信息服务层主要是指公共信息服务系统，通过 PML 服务器和 ONS 服务器提供 RFID 相关的公共信息服务。PML 服务器为企业应用软件提供 EPCIS(EPC information service)。EPCIS 使 RFID 相关信息以 PML 的格式来响应企业应用程序以及 ONS 要求。PML(physical markup language)即物理标记语言，是一种基于 XML，描述产品的动态数据和时序数据等详细信息的表达方式。PML 文件将被存储在一个 PML 服务器上，此服务器将配置一个专用的计算机，为其他计算机提供所需的文件。PML 服务器将由制造商维护，并且储存由这个制造商生产的所有产品的相关信息。

ONS(object naming services)即对象名解析服务，用来定位某一 EPC 对应的 PML 服务器。ONS 服务是联系前台中间件服务器和后台 PML 服务器的网络枢纽，并且 ONS 的设计与架构都以因特网域名解析服务(DNS)为基础，因此，可以使整个 RFID 网络以因特网为依托，迅速架构并顺利延伸到各地。RFID 标签中存储有产品的 EPC 码，RFID 中间件系统还需要根据这些 EPC 码匹配到相应的服装信息，这些服装的信息数据是整个分销管理系统所需要的基础数据。这个寻址功能就是由 ONS 提供的。

(4) 分销物流管理应用层

分销物流管理上层应用系统进一步对 RFID 采集和处理的信息进行分析处理，为成品仓库管理、配送管理和零售店管理等提供相应的支持，同时为 DRP(分销资源计划系统，用于管理分销网络)反馈成品仓库库存、配送指令执行情况、零售店进销存等信息，为 DRP 制定正确的配送指令提供支持。

2. 系统工作原理

基于 RFID 技术的分销物流管理系统首先接收 DRP 系统下达的配送计划或配送指令，根据分销过程监控模型对分销物流进行设置，构成基于 RFID 技术的分销物流监控环境；利用 RFID 数据自动集成技术从分销物流(包括仓库物流、配送物流、专卖店物流)中采集相应的物流数据；经相应的分析处理，为成品仓库管理、配送管理和零售店管理等提供相应的支持，同时为 DRP 系统反馈成品仓库库存、配送指令执行情况、零售店进销存等信息，为 DRP 制定正确的配送指令提供支持。

图 10-7 所示为基于 RFID 的分销物流管理系统的工作原理，其工作过程如下：

(1) 操作人员将 DRP 系统下达的配送计划或配送指令信息与其相关的产品关联起来，并根据分销过程监控模型对分销物流进行设置，构成基于 RFlD 技术的分销物流监控环境。

(2) 当附有 RFID 标签的产品进入到布置在分销各环节中的 RFID 读写器的有效读取范围时，读写器便将自动从分销物流(包括仓库物流、配送物流、专卖物流)中采集相应的物流数据，经 RFID 读写器的相应处理后，发送到其上层的 RFID 数据采集与处理系统。

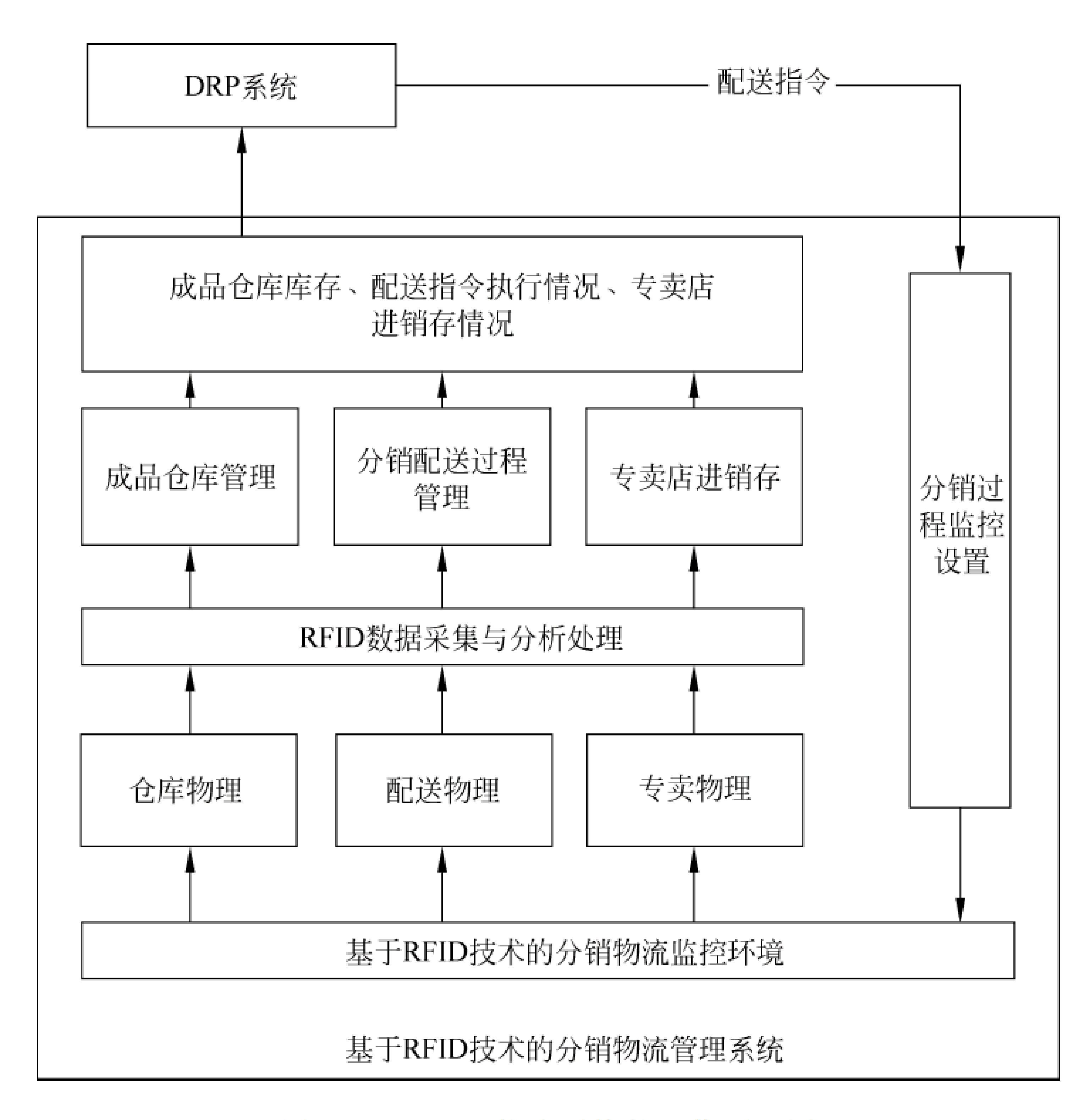


图 10-7　RFID 物流系统的工作原理图

(3) RFID 数据采集与处理系统依据产品生产物流模型,对由 RFID 读写器提供的数据进行各级处理,如将 RFID 原始数据转换成标签流(某个标签顺序经过多个 RFID 读写器的信息)、标签流转换成物流(某件商品顺序经过多个物理位置)、物流转换成信息流(如销售统计报表)等等,将 RFD 标签、读写器等与实物相关的内容还原成为信息流。

(4) 应用层进一步对 RFID 采集和处理的信息进行再次分析处理,为成品仓库管理、配送管理和专卖店管理等提供相应的支持,同时为 DRP 系统反馈成品仓库库存、配送指令执行情况、专卖店进销存等信息,为 DRP 制定正确的配送指令提供支持。

10.2.3　RFID 标签的安全隐患和解决方法

随着 RFID 标签在物流领域的广泛应用,与之相关的安全问题变得日益敏感。越来越多的商家和用户都在研究 RFID 系统在使用过程中如何确保其安全性和隐私性,防止个人信息和商业信息的泄露、复制和盗用。

RFID 标签的安全问题包括 3 个方面:即数据的秘密性、完整性和真实性。

(1) 数据的秘密性是指一个 RFID 标签不应当向未经授权的读写器泄露任何敏感的信息。尤其对于一些贵重的运输品、一些有毒的危险品和一些可能会危害到公共安全的运输物品,必须保证其所持 RFID 的信息具有保密性,防止这些物品的安全受到威胁。

(2) 数据的完整性是指 RFID 标签内所存的数据不能被随意篡改或替换。在基于公钥的密码体制中,数据完整性一般是通过数字签名来完成的。在 RFID 标签系统中,通常使用

消息认证码来进行数据完整性的检验。它使用一种带有共享密钥的散列算法，即将共享密钥和待检验的消息连接在一起进行散列运算，数据的任何细微改动都会对消息认证码的值产生较大的影响。如果不采用数据完整性控制机制，可写的标签存储器有可能受到攻击，信息可能被私自篡改甚至删除。

(3) 数据的真实性在 RFID 标签系统的许多应用中是非常重要的，其最重要的目的是为了防止伪造的标签欺骗读卡器。在商品应用模式，攻击者可以利用伪造的标签代替实际物品，或重写合法的 RFID 标签内容，使用低价物品标签的内容来替换高价物品标签的内容从而获取非法利益。同时，攻击者也可以通过某种方式隐藏标签，使读写器无法发现该标签，从而成功地实施物品转移。

针对上述安全隐患，RFID 标签通常采用的解决方法有以下几种。

(1) Hash 锁方式

由 MIT 提出的 Hash 锁是一种比较完善的抵制标签未授权访问的安全技术。整个方案只需要采用 Hash 函数，因此成本很低。标签验证阅读器原理如下：阅读器存储每个标签的访问密钥 K，对应标签存储着 metaID，其中 metaID＝Hash(K)。标签接收到阅读器的访问请求后发送 metaID 作为响应，阅读器通过查询获得与标签 metaID 对应的密钥 K 并发送给标签。标签通过 Hash 函数计算阅读器发送的密钥 K，检查 Hash(K)是否与 metaID 相同，相同则解锁，发送标签真实 ID 给阅读器。解锁过程如图 10-8 所示。

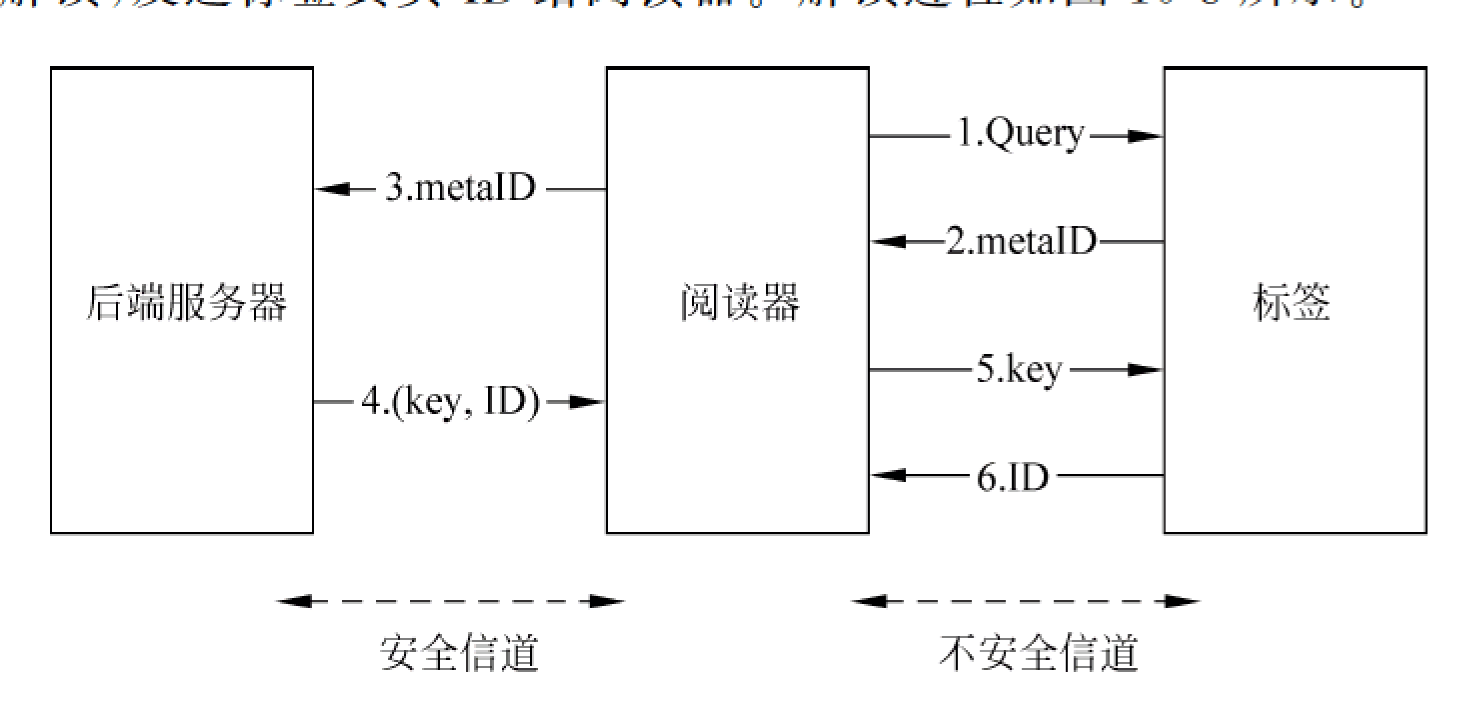


图 10-8　解锁过程图

该方案可以提供访问控制和标签数据隐私保护，但因其固定的 metaID，入侵者仍然可以通过 metaID 追踪标签，获得标签，定位隐私，并且访问密钥是以明文的方式通过前向信道传输，因此很容易被截获。

(2) 随机 Hash 锁方式

为了解决 Hash 锁中位置跟踪的问题，将 Hash 锁方法加以改进，采用随机 Hash 锁方法。随机 Hash 锁原理是标签包含 Hash 函数和随机数发生器，后台服务器数据库存储所有标签 ID。阅读器向标签 ID 发出询问，标签产生一随机数 R，计算 Hash(ID(R))，并将(R，Hash(ID(R))数据对传送给阅读器；阅读器收到数据对后，从后台数据库中取到所有的标签 ID 值，分别计算各个 Hash(ID(R))值，并与收到的 Hash(ID(R))比较，若 Hash(ID(R))相等，则向标签发送 IDk；若标签接收到的 IDk＝ID，此时标签即被解锁。在该方法中，标签每次回答是随机的，因此可以防止依据特定输出而进行的位置跟踪攻击。随机 Hash 法的缺点在于系统的效率不高和没有前向安全性。

(3) Hash 链

NLVR 实验室提出了一个 Hash 链方法，其保证了前向安全性。对于标签 ID，阅读器随机选取一个数 Si 发送给标签，并将(ID，Si)存储到后台数据库中，标签存储接收到 Si 后，进入锁定状态。当阅读器向标签发出询问消息时，标签回答 Ai＝G(Si)，并更新 Si＋1＝H(Si)，其中 G 和 H 为单向 Hash 函数，如图 10-9 所示。阅读器接收到 Ai 后，搜索数据库中所有的(ID，Si)数据对，并为每个标签计算 Ai′＝G(H(Si))，比较 Ai′是否等于 Ai，若相等，则返回相应 ID。该方案具有“前向安全性”，如果 Si＋1 被泄露，攻击者无法恢复出这之前的 Si，即秘密泄露之前标签的隐私都可以保证。但此方法运算量较大，不适合标签数目较多的情况。

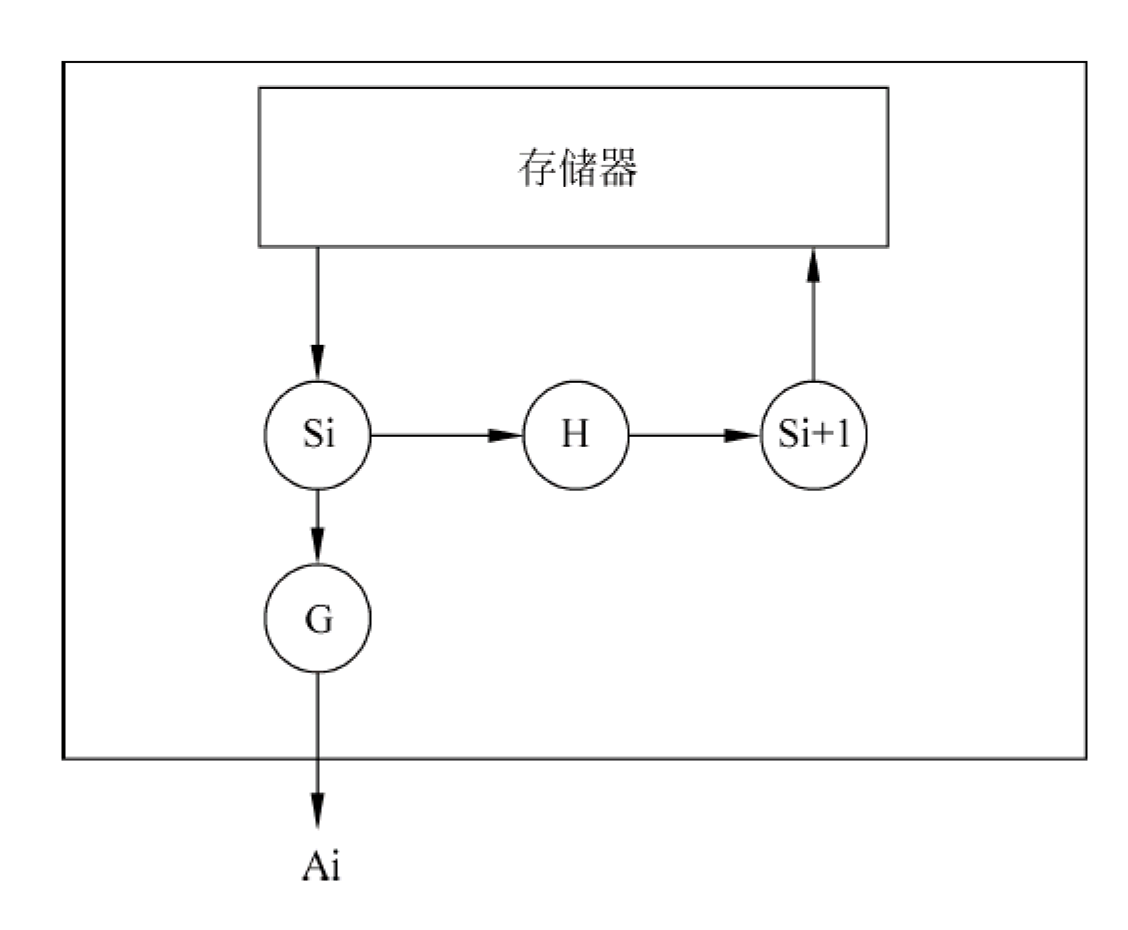


图 10-9　Hash 链原理示意图

以上各种方法对 RFID 的安全均进行了不同程度的改进，但是，任何仅依靠智能卡的安全性能来防范系统安全隐患的都是不完善的。一个完善的系统必须依赖于整体系统的安全性设计，利用各种可靠算法进行智能卡密钥的分散和智能卡数据的加密，并采取实时数据传输的手段来处理非正常数据，同时借助对嫌疑智能卡进行黑名单化等手段，最大限度地保障整个 RFID 系统的安全性能。

10.3　可信计算与智能卡

10.3.1　可信计算简介

可信计算是信息安全领域的一个分支。信息技术的高速发展、日新月异，为信息产业带来了空前的繁荣，极大地改善了人们的生活，但随之带来的安全问题不容忽视。

传统的安全防范技术是建立在被动防御思想之上，以防外为重点，如杀毒软件和防火墙。它们只对目前已知的恶意攻击和病毒有效，面对层出不穷的恶意攻击和花样翻新的病毒却无可奈何。可信计算技术的出现正是为了改变这一现状。

可信计算的基本思想是在计算机系统中首先建立一个信任根，再建立一条信任链，一级认证一级，一级信任一级，把信任关系扩大到整个计算机系统，从而确保计算机系统的可信度。所以说，可信计算技术是建立在主动防御思想之上，以确保内部安全为起点，由内向外

不断延伸。

可信计算机系统是可信计算平台的一个典型，它的信任链如图 10-10 所示。

信任链是通过构建一个信任根，从信任根开始到硬件平台、到操作系统、再到应用程序，一级一级地认证与信任，从而使这种信任关系延伸到整个计算机系统。可见，信任根的可信是整个系统的关键。目前，可信根是在主板上集成一个可信平台模块(trusted platform module，TPM)实现的。可信平台模块是一种安全芯片，由 CPU、存储器、I/O、加密协处理器、随机数产生器和嵌入式 OS 等部件组成，如图 10-11 所示，与 CPU 智能卡芯片十分相似。

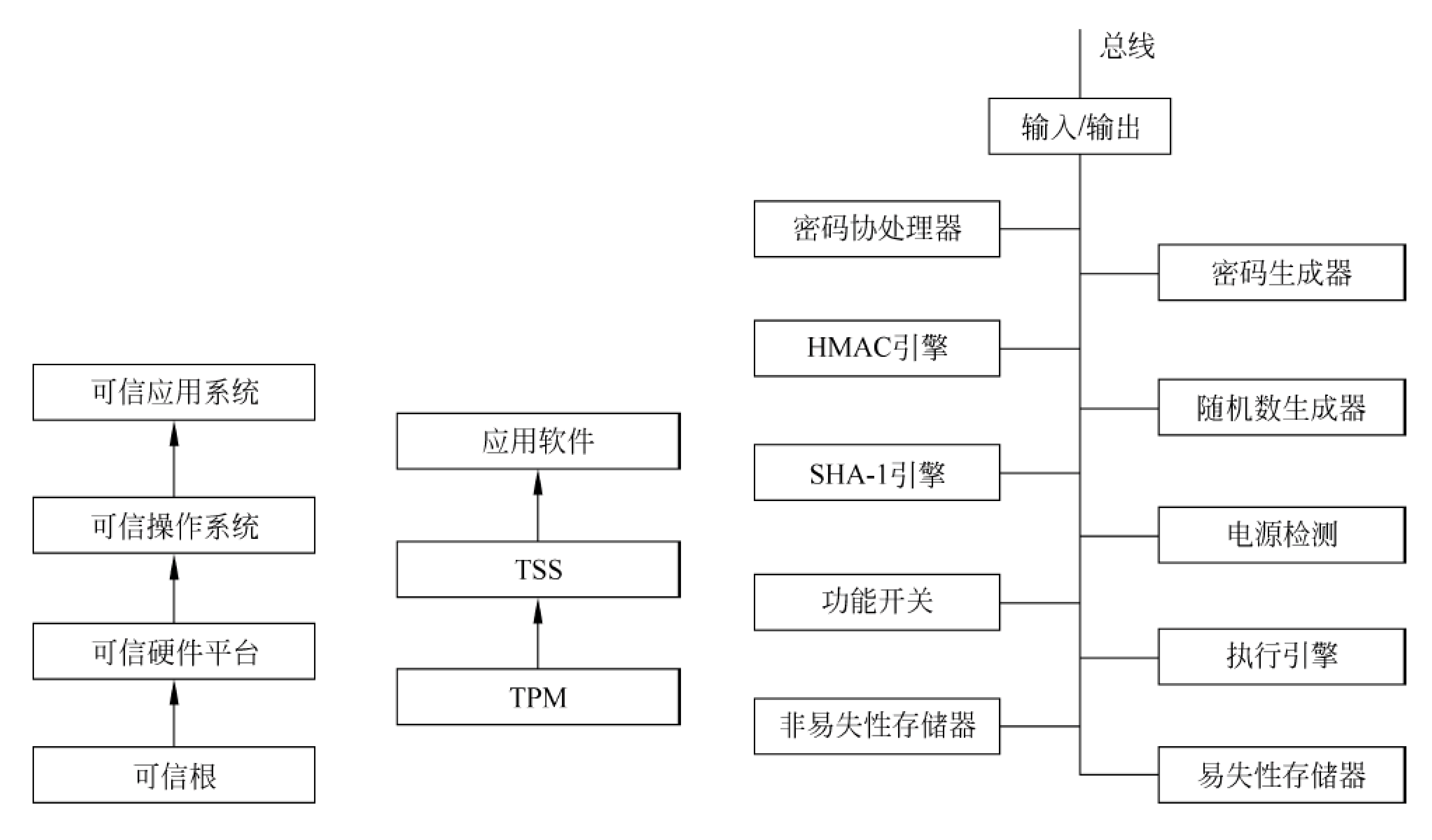


图 10-10 信任链与可信平台结构示意图

图 10-11 TPM 体系结构图

可信计算软件栈(TCG software stack，TSS)位于应用软件和 TPM 之间，如图 10-10 所示，是应用软件和 TPM 的接口，为开发基于 TPM 的可信程序提供软件接口。所有加密、签名、完整性测量(根据摘要值测量应用软件或敏感信息是否完整、是否被篡改)和报告等功能都是在 TPM 内部完成，TSS 负责建立应用程序与 TPM 的会话，完成应用程序对 TPM 硬件资源的合法调度。

可信计算涉及的研究和应用领域非常广，TPM 与 TSS 仅是围绕可信终端简单模型的技术，目前还有面向可信网络的可信网络链接(trusted network connect)、可信服务器等技术。可信计算作为一门新起的技术，它的理论、关键技术和具体应用的解决方案设计目前还不十分完善，有待科研工作者的进一步研究。

国际上关于可信计算最有影响力的组织是可信计算工作组(Trusted Computing Group，TCG)，于 2003 年成立，前身是可信计算平台联盟(Trusted Computing Platform Alliance，TCPA)，由 Intel、Compaq、HP、IBM 以及 Microsoft 等公司组成。TCG 在原 TCPA 强调安全硬件平台构建的宗旨之外，更进一步增加了对软件安全性的关注，旨在从跨平台和操作环境的硬件组件和软件接口两方面，促进与厂商独立的可信计算平台工作标准的制定。TCG 目前发布的标准主要有以下几个方面：

(1) TCG体系结构总体规范(Architecture Overview)。

(2) 基础框架规范(Infrastructure Specifications)。

(3) 可信平台模块规范(Trusted Platform Module Specifications)。

(4) 软件栈规范(TSS Specifications)。

(5) 可信网络连接规范(Trusted Network Connect Specifications)。

(6) 个人计算机客户端规范(PC Client Specifications)。

(7) 服务器规范(Server Specific Specifications)。

目前TCG所提出的规范还没有成为国际标准化组织的标准。从国家安全的角度出发,对我们来说符合TPM规范的芯片也不一定可信,所以研究并推出我国可信计算标准是势在必行的。发展可信计算技术是涉及国家安全和国家利益的一件大事,同时还可以节省我国软硬件产品开发商向TCG支付的巨额可信授权和评估费用,并促进民族工业的发展。中国可信计算联盟在2008年成立,由企事业单位、科研单位、相关用户和个人组成,宗旨是:"以企业为主体,产学研用联合,促进我国可信计算产业链的形成和发展,增强企业竞争力"。中国可信计算联盟将协调产业链促进可信计算标准出台,为我国可信计算的发展引领航程。

10.3.2 智能卡在可信平台中的应用

TCG所提出的TPM是集成在计算机主板上,目前看来这种设计方法有两个弊端:一方面,没有集成TPM的计算机由于缺少这一至关重要的信任根,无法获得面向终端的可信计算,在网络中它是不可信的,而现阶段大部分计算机主板都没有集成TPM模块,短时间内全部更新成嵌入TPM的计算机是不可能的。另一方面,TCG认为可信的计算机系统对中国来说不一定可信,很有可能留有后门成为以后信息战的突破口,所以中国的可信计算机必须采用中国可信的根。针对上述两个弊端,有学者提出了采用一种基于USBKey的外挂可信模块来替代TPM的解决方案。

USBKey是现在已成熟使用的加密安全模块,内部集成了CPU智能卡芯片,具有运算功能,适用于敏感、高安全性场合,是智能卡的一个重要分支。USBKey不同于传统的接触式、非接触式智能卡,其大多是充当密码服务提供者的角色,采用USB接口可以安全、方便、快速地与电子设备进行数据交换。如10.3.1节所述,TPM的内部体系结构与智能卡非常相似,而且和TPM私钥永不会被导出一样,USBKey的私钥也是安全存储的,这就使得基于USBKey的外挂可信模块的实现成为可能。同时USBKey中还可以载入我国自主密码算法,真正做到可信根被中国可信。

上面所说的可信平台都是可信计算机系统,可信计算理论最初就是在研究计算机系统安全时建立的。随着技术的发展,可信计算技术也被应用到可信嵌入式终端设备的研究中,是解决嵌入式终端安全性问题的一个有效手段。可信嵌入式终端设备中信任根的解决方案是一个关键问题,同可信计算机系统相似,也可以将TPM集成到嵌入式设备中,但是集成式TPM需要修改原有的嵌入式硬件体系结构,成本较高,难以推广,所以采用外挂便携式TPM是一种方便、低廉、有效的方案,便携式TPM可以采用USBKey的形式与嵌入式主机相连。

以上分析的案例都是采用USBKey作为外挂式TPM,取代了原有TPM芯片作为可信根,看起来TPM与智能卡是竞争关系,其实不然,在一些场合智能卡与TPM还可以协同工

作。相比 TPM,智能卡具有便携性,但是容量较小,当两者协同工作时,可以提供一个更加强大、安全的环境。现在的 EFI(extensible firmware interface)BIOS 虽然比传统的 16 位 BIOS 更先进,但是启动项管理缺少保护机制,而且可以随意查看硬盘资料,并不符合可信平台规范,于是有学者提出了一种采用 USBKey 增加 BIOS 的安全控制、增强日志及密钥安全管理的方法,可以解决 PC 系统底层安全问题。TPM 可以保证计算机终端可信,但是不能保证使用终端的用户可信。一种方法就是设置终端登录的 PIN 码,可信的计算机可以保证 PIN 码不被病毒盗取,但是不能保证 PIN 码其他方面的脆弱性(如大多数人会将生日或电话作为 PIN 码)。而智能卡有身份认证的功能,可以保证登录用户身份的真实性,使得计算机终端和用户都是可信的,所以将智能卡的身份认证技术应用于可信计算机系统是十分必要的;对于非常敏感数据的生成或高度安全的操作,可以采用二次签名来实现,第一次由可信计算机签名,第二次由智能卡签名,这就可以证明该数据或操作是值得信任的人在可信的环境(计算机)中完成。

目前还有一种前沿技术,是利用 Java 智能卡完成 PC 应用软件的可信量度。由于可信计算的核心是信任链的传递,可信量度技术是保证信任链传递的关键。但是现在复杂操作系统无法被有效量度,所以信任链无法传递到应用程序。针对这个难题,有学者提出了将可信量度机制建立在独立于 PC 不可信平台、基于虚拟技术、面向关键应用的 Java 智能卡可信环境中,大意是将应用程序安全下载到 Java 智能卡中,在 Java 智能卡中完成可信量度,并将结果密文传输回 PC。

图 10-12 对本单元智能卡在可信平台中的应用做了小节,图中的所有智能卡很难在一个系统中同时出现,但通过此图,读者可以清晰地发现智能卡在可信计算平台中有着广泛的应用前景。

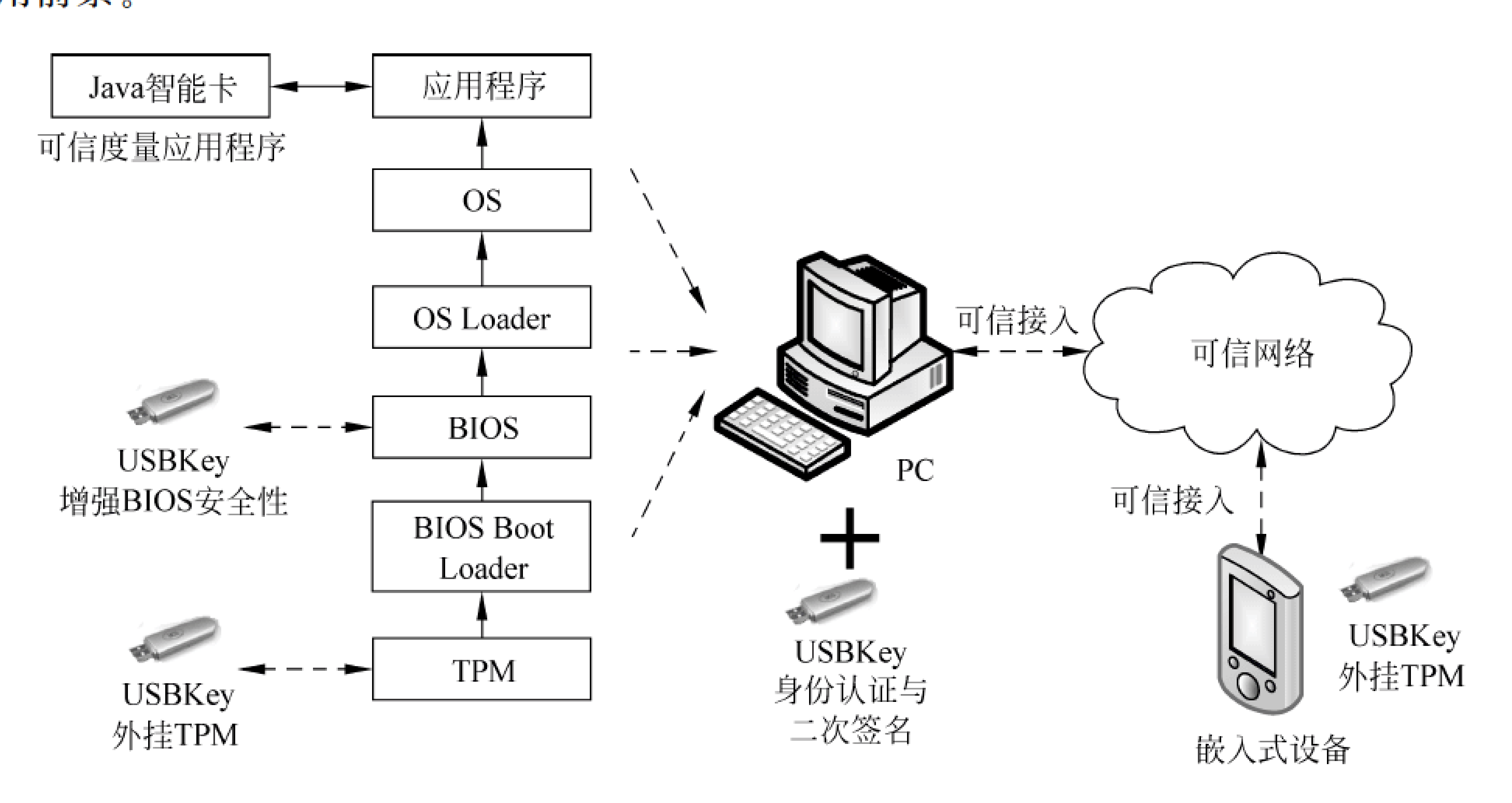


图 10-12 智能卡在可信平台中的应用

10.3.3 可信计算思想在智能卡中的应用

10.3.2 节讲述了智能卡在可信计算技术中发挥的作用,智能卡利用其自身高安全性、便携简单、身份认证和与 TPM 硬件相似的特点,可以扩展信任根实现的方法、改进可信量

度的实现方案,同时还可以增强可信计算系统的安全性和可行性。可以说,将智能卡技术引入可信计算领域,将推动可信计算技术的发展和完善。本节将介绍把可信计算思想引入智能卡技术中,看看可信计算思想是如何帮助智能卡提高自身安全性的。

纵观智能卡安全技术的发展,可以发现现有的防克隆、防篡改等技术都是建立在被动防御的思想之上,先有攻击者发现漏洞,后才有人发明防御技术,可见智能卡安全防御技术通常都是滞后于攻击技术的。但是可信计算思想则不同,先建立自身完善的安全体系,让攻击者无从下手,这样不但可以极大地提高智能卡自身的安全性,减少因安全漏洞带来的经济损失,还可以改变现有智能卡安全防御技术的被动姿态,扭转当前防御者在与攻击者博弈中的不利形势。

如何将可信计算思想引入到智能卡技术中?这个课题目前还在前沿分析阶段,没有被认可的成熟案例。将可信计算引入智能卡,主要是研究基于智能卡的 TPM 与 TSS 的设计。

可信计算的基础是可信根的可信,TPM 的设计与实现是关键。智能 CPU 卡的硬件结构和 TPM 非常相似,都需要易失/非易失性存储器、对称/非对称加密协处理器、随机数发生器、杂凑运算单元和执行单元。不同的是 TPM 需要监测电源,RSA 密钥长度需达到 2048 位。而且由于 TPM 是集成在主板上,它的 I/O 接口是总线,而智能卡可能是 7816 接口、射频接口或 USB 接口等。可见,TPM 与智能 CPU 卡硬件的组成结构基本相同,几处不同之处并不会对将智能 CPU 卡当作 TPM 使用产生原理性的矛盾。所以,选择一款合适的智能 CPU 卡芯片,在硬件上就可能满足可信根 TPM 的需要。当然,在现有智能 CPU 卡的基础上开发一款功能更强大的、更符合可信计算平台需要的嵌入式安全芯片也是非常好的解决方案。

根据 JavaCard 结构(参见本书第 12.1.2 节),连接底层硬件与 JavaCard 应用程序 Applet 的是本地方法集、JavaCard 运行环境、JavaCard 虚拟机和 JavaCard API 库函数。可以将 BootLoader 作为信任起点,上电后负责量度硬件平台和运行环境的完整性。之后运行环境占有 CPU 的控制权,信任链传递到运行环境,由它负责量度 JavaCard 虚拟机和 API 库函数以及其他组件的完整性。运行环境还可以对下载到本地的 Applet 进行量度,使 JavaCard 只运行通过本地量度的 Applet。原有的 JavaCard 技术规范并没有规定可信计算在智能卡应用的相关技术,所以我们需要对 JavaCard 虚拟机、JavaCard API 库函数和 JavaCard 运行环境进行修改完善以达到可信平台的要求,还需要在非易失性存储器中开辟空间作为可信存储根和可信报告根。

将可信计算思想引入智能卡是一个热门课题,但是目前此项技术还没有完善的理论和相关规范,也没有被认可的产品,还需要各条战线上的科研工作者共同努力,推动智能卡安全技术的发展。

10.4 电子商务中的 USBKey 身份认证

电子商务(E-commerce,EC)是指个人或企业通过 Internet 网络,采用数字化电子方式进行商务数据交换和开展商务活动的贸易运作模式。随着 EC 的不断普及,其面对的威胁也更加多种多样,人们对身份认证的安全性要求越来越高。

10.4.1 电子商务中的威胁与USBKey对策

电子商务与常规商务活动的区别在于电子商务要依托于网络，而由于网络的操作不需要进行面对面地交流，所以身份的验证成为了一个关系到交易安全的重要问题。电子商务的安全威胁主要来自以下几个方面。

(1) 攻击者盗用合法用户的身份信息，以仿冒的身份与他人进行网上交易或网上支付。

(2) 攻击者在网络的传输链路上，通过物理或者逻辑的手段，对数据进行非法的截获与监听，从而获得通信中敏感的秘密数据信息；或者对截获的数据内容进行篡改(增加、截取或者修改)，从而破坏消息的机密性和完整性。

(3) 某些用户可能对自己发出的信息进行否认，冒充其他用户进行“信息的伪造”，把伪造的交易信息进行发送，破坏消息的真实性。

(4) 攻击者进行重放攻击。攻击者截获网络上的密文信息后，并不将其破译，而是把这些数据包再次发送，以实现恶意攻击的目的。

随着硬件技术的飞速发展，USBKey的出现提供了一种很好的解决方案，它集数据加密和数据存储两大功能于一身，目前已经成为了众多银行和主要电子商务企业的首选安全认证解决方案。

USBKey产品最早是由加密锁厂商提出来的，原先的USB加密锁主要用于防止软件破解和复制，保护软件不被盗版，而USBKey主要用于网络认证，其内部主要保存数字证书和用户私钥。USBKey是一种USB接口的硬件设备，它内置单片机或智能卡芯片，有一定的存储空间，可以存储用户的私钥以及数字证书，利用USBKey内置的公钥算法实现对用户身份的认证。由于用户私钥保存在密码锁中，理论上使用任何方式都无法读取，因此保证了用户认证的安全性。

每一个USBKey都具有硬件PIN码保护，PIN码和硬件构成了用户使用USBKey的两个必要因素。用户只有同时取得了USBKey和对应的用户PIN码，才可以登录系统。即使用户的PIN码被泄漏，只要用户持有的USBKey不被盗取，合法用户的身份就不会被仿冒；如果用户的USBKey遗失，拾到者由于不知道用户PIN码，也无法仿冒合法用户的身份。USBKey严格按照ISO/IEC 7816-4标准设计，能够安全地完成密码算法的存储和执行，以及敏感数据的保存等功能，它一般体积比较小，可以像钥匙一样随身携带，同时可以实现即插即用的功能，因此可以像普通钥匙一样使用，非常方便。

USBKey内置散列算法(如MD5、SHAx)和随机数生成器，它可以被预置一个密钥或存入数据证书，来确定用户的身份。访问者首先需要把USBKey插入USB接口中，输入一个PIN值，进一步证实自己的身份。如果身份认证服务器不能识别访问者的身份，网络访问将被拒绝。这样可以使网络访问者经过双重认证，才能获得网络访问授权，网络认证的安全性得到很大的加强。另一方面，USBKey不仅能够提供智能卡所能提供的所有安全机制，而且价格比智能卡要低，它无需昂贵的智能卡阅读器，大大降低了成本。

USBKey身份认证主要有如下两种应用模式：

(1) 基于挑战-响应认证模式

USBKey内置单向散列算法，预先在USBKey和服务器中存储一个证明用户身份的密钥，当需要在网络上验证用户身份时，先由客户端向服务器发出一个验证请求。服务器接到

此请求后生成一个随机数,回传给客户端PC上插着的USBKey,此为“挑战”。USBKey使用该随机数与存储在USBKey中的密钥进行MD5运算,得到一个运算结果作为认证证据传送给服务器,此为“响应”。与此同时,服务器使用该随机数与存储在服务器数据库中的该客户密钥进行MD5运算,如果服务器的运算结果与客户端传回的响应结果相同,则认为客户端是一个合法用户。图10-13表示USBKey、PC和网络服务器之间的相互认证过程。

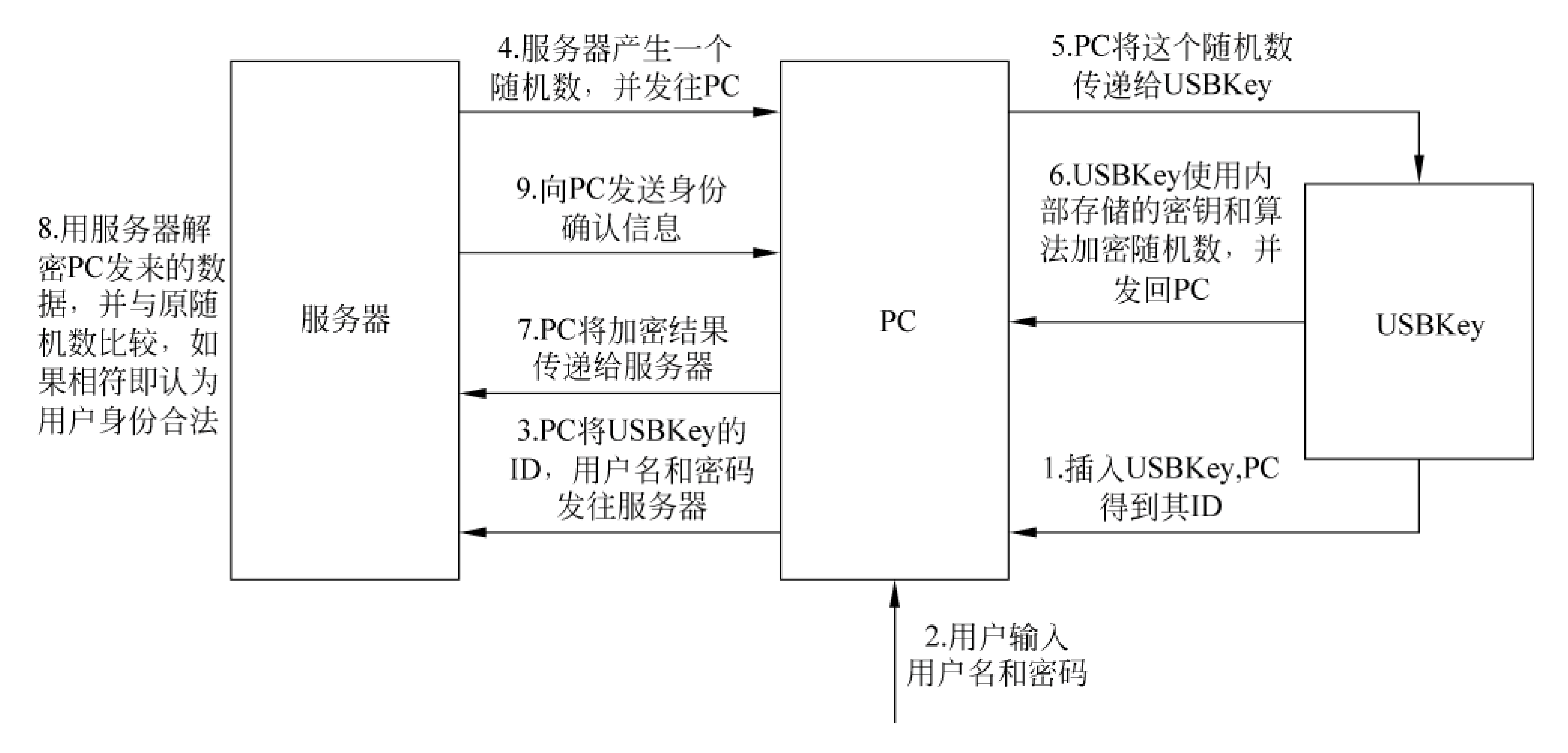


图10-13 认证过程示意图

(2) 基于PKI的数字证书认证模式

PKI利用一对互相匹配的密钥进行加密、解密,包括一个公共密钥(公钥,public key)和一个私有密钥(私钥,private key)。其基本原理是:由一个密钥进行加密的信息内容,只能由与之配对的另一个密钥才能进行解密。公钥可以广泛地发给与自己有关的通信者,私钥则需要十分安全地存放起来。每个用户拥有一个仅被本人所掌握的私钥,用它进行解密和签名;同时拥有一个公钥用于文件发送时加密。当发送一份保密文件时,发送方使用接收方的公钥对数据加密,而接收方则使用自己的私钥解密,这样,信息就可以安全无误地到达目的地了。即使信息被第三方截获,由于没有相应的私钥,也无法进行解密。

挑战-响应模式可以保证用户身份不被仿冒,但无法保证认证过程中数据在网络传输过程中的安全。而基于PKI的“数字证书认证方式”可以有效保证用户的身份安全和数据传输安全。数字证书是由可信任的第三方认证机构——数字证书认证中心(CA)颁发的一组包含用户身份信息(如私有密钥)的数据文件,PKI体系通过采用加密算法构建了一套完善的流程,保证数字证书持有人的身份安全。而使用USBKey可以保障数字证书无法被复制,所有密钥运算在USBKey中实现,用户密钥数据不在计算机内存出现也不在网络中传播,只有USBKey的持有人才能够对数字证书进行操作。由于USBKey具有安全可靠、便于携带、使用方便、成本低廉的优点,加上PKI体系完善的数据保护机制,使用USBKey存储数字证书的认证方式已经成为目前主要的认证模式。

10.4.2 体系结构及认证过程

USBKey的整个体系结构由硬件层、核心驱动层、标准中间件层和应用层构成,如图10-14所示。

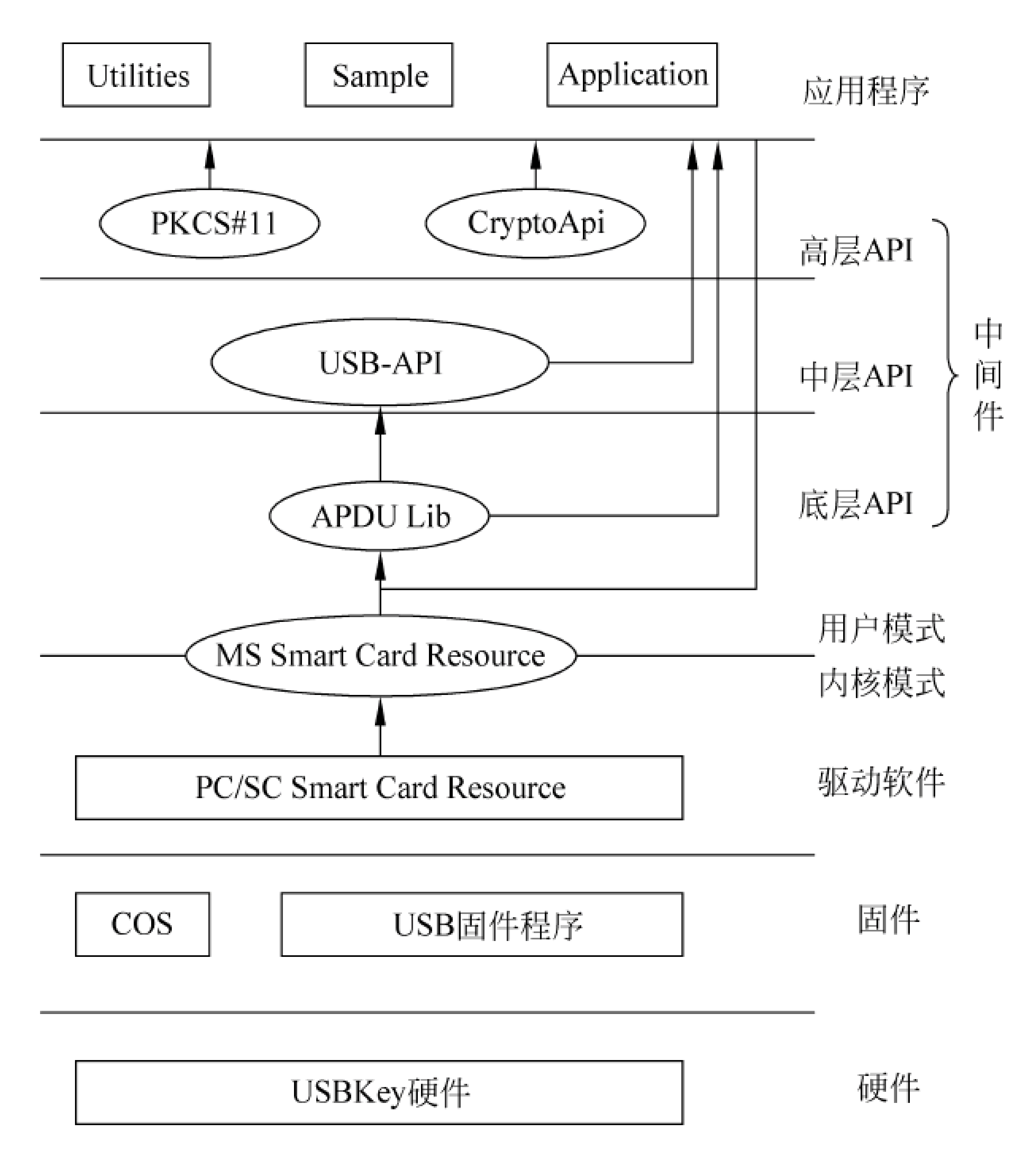


图 10-14　USBKey 体系结构图

应用软件是指针对 USBKey 开发的各类应用，如网络登录软件或者文件加密器等。

中间件层处于应用层和设备驱动之间，包括基于具有跨平台特性的 PKCS#11 标准接口(详细参见第 7 章)和 Windows 平台的 CSP 接口(详细参见第 9 章)，应用开发者无需移植，可方便使用。

驱动软件是主机端的 USB 驱动程序，是依据微软定义的 PC/SC 标准开发的驱动接口，使得上层可以通过 Win32 标准函数集访问 USBKey。该层负责协调用户主机与硬件层之间的数据交互操作和处理上层应用对 USBKey 的访问请求。

USBKey 内置 USB 控制器、微处理器、存储器。USB 控制器负责识别 PIN 码，以及整个 USBKey 的驱动；存储器中存放已签名的数字证书和用户的认证信息(如 PIN 码)，其中数字证书存储之后不能修改，认证信息在初始化之后，系统只能对其进行读操作，不能任意改动，从而保证信息的安全存储；微处理器负责处理 Hash 函数、加密、解密函数的计算和控制。在使用时，USBKey 通过 USB 接口和计算机连接，实现信息的安全传输。

USBKey 身份认证和权限校验基本流程如图 10-15 所示。

流程的具体描述如下，括号内标注的数字对应图 10-15 中的步骤。

(1,2) 用户通过安全中间件(USBKey)向系统提出认证申请。插入 USBKey 弹出 PIN 口令，用户输入正确的口令认证方可继续，当输入错误口令超过一定次数，用户密码将被锁住，此时必须由管理员来解锁用户密码。

(3,4) 认证服务器从 USBKey 中得到用户的数字证书，发送给 UIAS 服务器进行查验，从中获得用户信息(身份、角色)。

(5) 当身份信息验证成功时返回认证结果和用于数字签名的随机数。

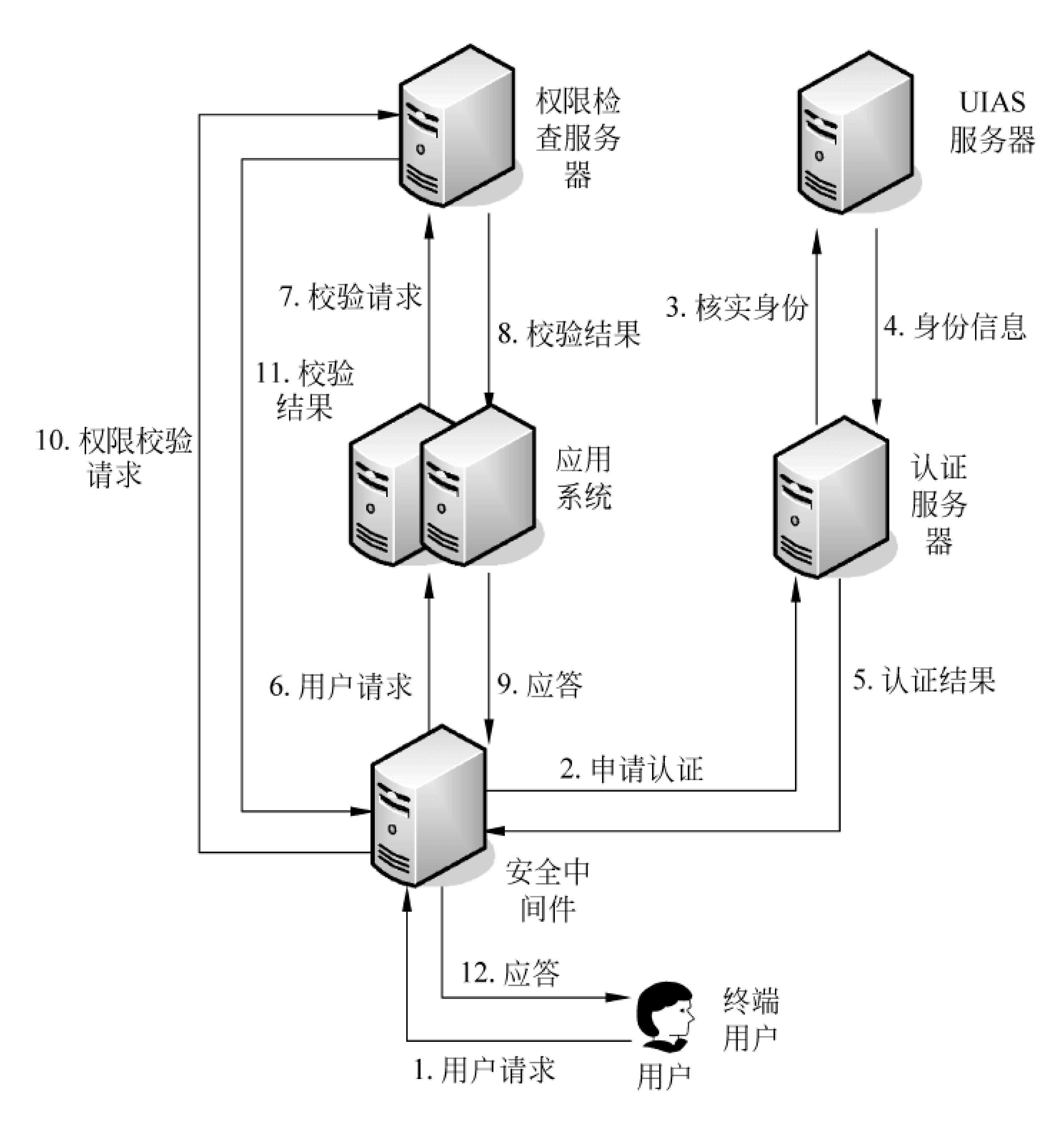


图 10-15 **身份认证与权限校验流程图**

(6) USBKey 对自己产生的随机数和服务器返回的随机数进行数字签名,发送签名信息以及用户证书到应用系统服务器,进行权限校验。

(7) 应用系统将权限校验请求发送给权限检查服务器,验证用户证书的合法性和有效性,合法性通过与相应证书管理数据库进行匹配验证,验证有效性是指验证该证书是否过期。

(8,9) 若以上签名和证书的认证检验都成功则此用户被赋予相应的权限并返回应答。

(10,11,12) 用户身份被认证后,用户可以向权限校验服务器直接发出访问控制的请求,也可以通过应用系统向权限校验服务器发出访问控制的请求。权限校验服务器返回校验结果,此时用户可以访问应用系统。

10.4.3 系统安全分析

USBKey 系统采用了公钥密码体制,遵从 PKI 相关的技术标准。PKI 是一种用非对称密码算法原理和技术实现并提供安全服务的具有通用性的网络安全基础设施,是一种遵循标准的利用公钥密码技术为电子商务、电子政务开展一整套安全的基础设施。它采用证书管理公钥,通过第三方可信机构 CA 把用户的公钥和用户的其他标识信息(如名称、身份证号)捆绑在一起,实现对用户身份进行安全认证的目的。

PKI 这种遵循标准的密钥管理平台,能够为所有网络应用提供采用加密和数字签名等密码服务所需要的密码和证书管理。PKI 是一种基于公钥密码技术,通过数字证书建立信任关系,保证网络通信安全的技术。在 PKI 体系中,CA 和数字证书是密不可分的两个部

分。CA负责产生、分配并管理数字证书，其通常采用多层次的分级结构，上级负责签发和管理下级的证书，而最下一级的认证中心直接面向最终用户。用户向CA申请证书，申请时用户的申请信息都将从申请程序传到系统的加密程序中。加密程序会将正确的数据传送到名为加密服务提供程序(CSP安装在用户的计算机上)的程序中，它将在用户的计算机上生成一个公钥和一个私钥，这两个密钥通常被称为密钥对。密钥生成后，软件CSP将进行加密并保护私钥的安全。公钥连同证书申请者信息一起被发送到证书颁发机构。一旦CA根据它的策略确认了证书请求，它将使用它自己的私钥在证书上创建数字签名，然后将它颁发给申请人。随后证书申请人将获得来自CA的证书，以便将它安装在计算机的适当证书存储区。

用户可以将该数字书证书作为用户认证的凭据。证书中的数据包括了来自证书主题的公钥和私钥对中的公开加密密钥。对于用发送方的私钥签署的消息，消息接收方可以用发送方的公钥验证其真实性。该密钥可以在一份发送方的证书中找到。使用证书上的公钥来验证签名，可以证实签名是否是使用证书主题的私钥生成的。如果发送方一直很好地保持私钥的机密性，接收方就可以相信消息发送方的身份。

电子证书的结构和身份验证协议采用国际标准X.509。X.509身份认证协议是一种基于PKI的非对称密码协议，该协议采用X.509证书和非对称密码技术(即公钥密码技术)进行身份认证。X.509身份认证协议提供了三种认证方案：单向身份认证、双向身份认证和三向身份认证。下面介绍一下使用最多的双向认证。双向认证流程如图10-16所示。

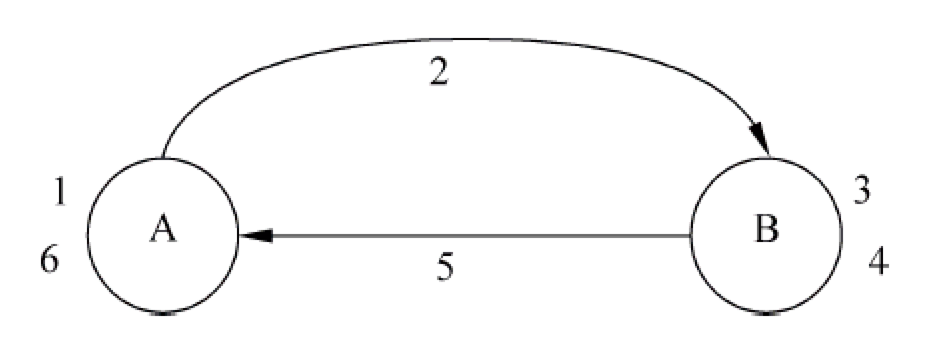


图10-16　双向身份认证图

流程的具体描述如下：

(1) A生成一个非重复的随机数r_A，用来抗重放攻击。

(2) A向B发送消息：A{t_A,r_A,B}。

(3) B受到消息后执行以下动作：获取A的证书，并验证证书的有效性。从A的证书中提取公钥，验证A的签名，同时检验消息的完整性。检查B自己是否是消息的接收者。验证时间戳t_A是否为当前时间。检查r_A是否被重放。

(4) B生成一个非重复的随机数r_B，作用与r_A相同。

(5) B向A发送消息：B{t_B,r_B,A,r_A}。

(6) A收到消息后执行以下动作：获取B的证书，并验证证书的有效性。从B的证书中提取公钥，验证B的签名，同时检验消息的完整性。检查A自己是否是消息的接收者。验证时间戳t_B是否为当前时间。检验r_B是否被重放。

X.509身份认证协议是利用非对称密码技术实现的，它的安全性从根本上取决于所使用私钥的安全性。这包括CA私钥的安全性和用户私钥的安全性。一旦CA的私钥泄密，所有由其颁发的证书的安全性都将受到威胁。窃取CA私钥的非法入侵者可伪造用户证书，冒充成合法用户。因此保证CA私钥安全至关重要。通常CA的私钥是由加密机生成并保存在加密机中，由加密机提供安全机制来保证私钥的安全。用户私钥的安全也很重要。如果用户私钥被窃取，非法入侵者就可以伪造用户的签名，从而使X.509身份认证失败。私钥泄密的途径有两个：一是CA把私钥传送给用户的过程中被截获，二是私钥在使用过程中被窃取。因此要保证私钥的安全，就必须保证私钥在上述两个过程的安全。将电子证

书存储于 USBKey 中可以解决上述问题,但是当非法者取得了 USBKey 和与之配套的 PIN 密码后,以上的所有安全都不复存在,这是常规的 USBKey 所无法解决的,所以将生物识别技术比如指纹识别技术与 USBKey 相结合成为了必然趋势。但成本问题成为了制约其发展的主要原因。

10.5 3G 系统中 USIM 的应用

10.5.1 移动通信 USIM 卡

移动通信是智能卡的主要市场之一,是智能卡的重要应用技术领域,通信卡始终在智能卡中占有很高的市场份额。通信技术快速发展的大背景,也为移动通信智能卡的发展提供了大舞台。

随着无线通信技术的发展,人们从模拟制式的蜂窝移动通信系统进入到数字通信时代,智能卡作为网络注册载体被广泛应用于无线通信领域,以 GSM 和 IS-41 为代表的第二代移动通信系统解决了第一代移动通信中存在的频谱利用率低,业务种类受限,通话易被窃听的技术缺陷。但随着业务种类的不断增多,安全需求和服务质量需求的不断提高,二代移动通信系统已经不能满足需求。GSM 系统只能进行网络对用户的认证,而不能进行用户对网络合法性的认证,用户容易受到假冒基站的攻击,且认证和密钥的生成算法现在已在因特网上传开。因此随着网络技术、数据和多媒体通信技术的发展,第三代移动通信系统应运而生。它不仅可以提供多种类型、高质量的多媒体服务,实现全球无缝覆盖,还可以为用户提供高速的数据业务,以及诸如电子商务、因特网服务等业务。更重要的是第三代移动通信网络的安全接入,添加了用户认证机制,设计了新的认证和密钥协商协议,可以实现用户和网络之间的双重认证,大大加强了无线接入的安全性。此外,第三代移动通信网络加强了加密算法,并增加了数据的完整性保护。

USIM 卡作为智能卡中的一种,具有写入数据和存储数据的能力,卡存储器中的内容根据需要可以有条件地供外部读取,供内部信息处理和判定之用。USIM 卡属于应用于电信领域的微处理器卡,卡中的集成电路包括处理器(CPU)、EEPROM、随机存储器(RAM)以及固化在只读存储器(ROM)中的卡操作系统(COS)和加密辅助电路等。

在详细讨论 USIM 卡之前,首先看看其与 SIM 的关系与区别。从 SIM 卡开始使用到当前,主流的 SIM 卡都使用 8 位的 CPU,通常是 Intel8051 或者 Motorola6805,部分高档的 JavaCard 一般采用 32 位结构。卡中的内存数据由最初的 256 字节的 RAM 和 3K 字节的 EPROM,到现在 RAM 已经增长到 1K 字节甚至更多,EEPROM 增长到了 128K 甚至 144K 字节。SIM 卡使用底层基础软件作为卡内操作系统,把相对成熟的应用程序和函数固化在 ROM 中,数据文件保存在 EEPROM 中,而 RAM 主要被用作卡与手机之间通信的数据缓冲区。智能卡的逻辑结构如图 10-17 所示。

从 20 世纪 90 年代中期开始,随着更多的新应用出现,SIM 卡逐渐开始演变为功能更加强大的应用平台。软件架构逐渐向虚拟机方向(尤其是 Java 虚拟机)发展,硬件在计算能力和存储容量上有很大提高。从体系结构来看,可以把电信智能卡分为三代。第一代电信智能卡只包含纯粹的 GSM 认证功能。第二代电信智能卡中引入了主动式命令——卡应用工

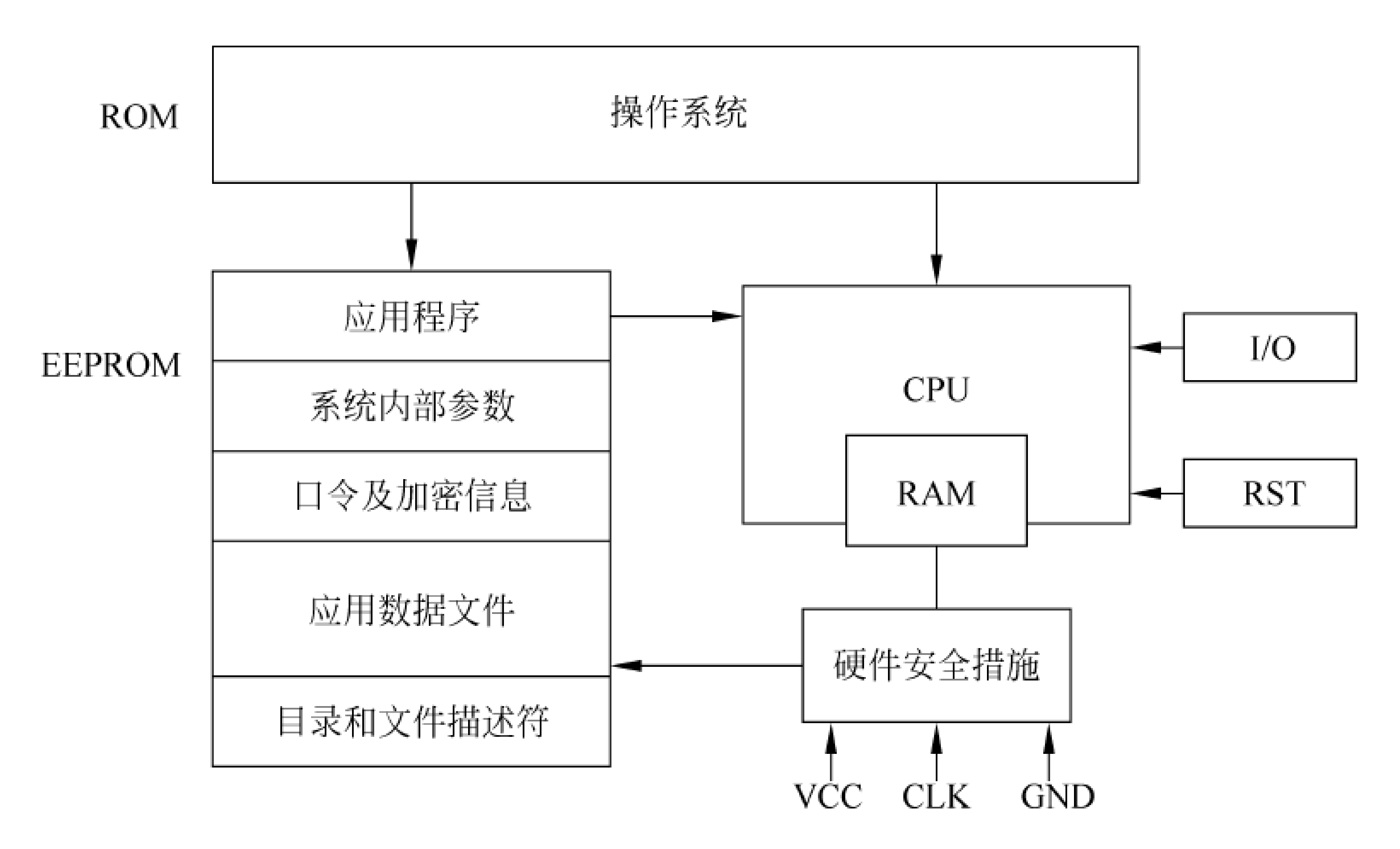


图 10-17 智能卡逻辑结构图

具包，可以通过 SIM 卡应用工具包 STK 开发其他的增值应用。第三代电信智能卡中引入了 UICC(通用集成卡电路)平台和多应用的概念。GSM 认证、USIM 认证成为平台上处于同等地位的不同应用。三代电信智能卡的体系结构比较如图 10-18 所示。

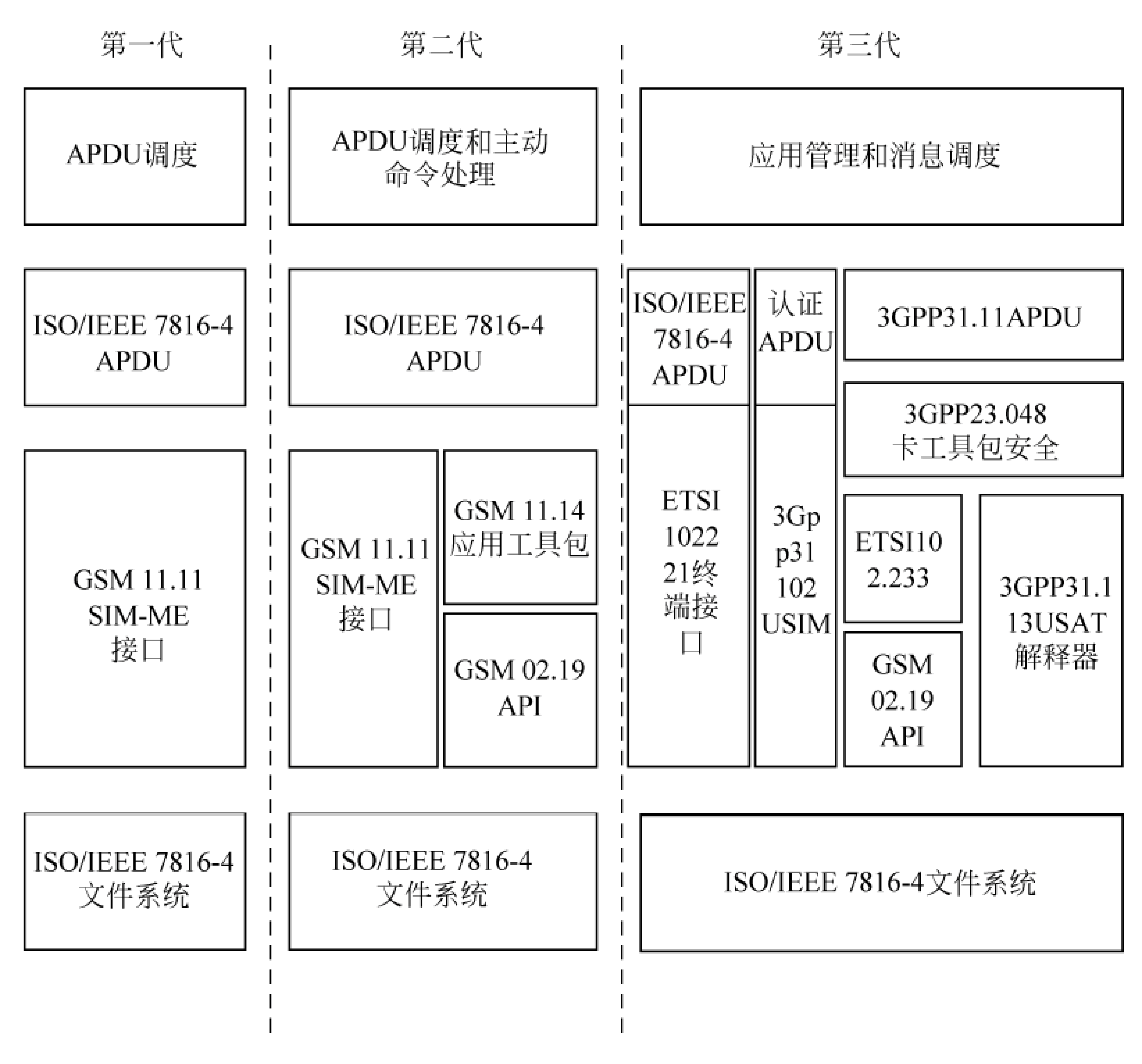


图 10-18 三代电信智能卡体系结构对比

SIM 是 2G 定义的 ICC。它最初被规定为一个物理和逻辑上统一的实体，不对平台和应用加以区分，SIM 既是卡平台又是 GSM 应用。UICC 则是 USIM 物理和逻辑上的平台。

它除了支持 USIM 应用、GSM 应用之外，还可以支持其他非电信应用。USIM 是 UICC 上的 UMTS 应用。USIM 与 SIM 不同，它不是一个物理实体，而完全是 UICC 上的一个逻辑应用。它仅接受 3G 指令，与 2G ME 不兼容。USIM 可以提供机制支持进行 2G 的鉴权和密钥协商以使 3GME 可以访问 2G 网络。在 GSM 网络中，智能卡有非常单一的用途，即提供安全机制以保护敏感的网络数据并进行鉴权操作，同时为应用的开发和部署提供平台。在 3G 网络中，智能卡引入多应用的概念，它能够同时处理多个任务并服务于多个应用。USIM 卡物理特性与 SIM 卡基本相同，但数据处理能力和存储容量比 SIM 卡强得多，而且拥有更快的硬件和更高的安全性能。

电信领域的 CPU 卡在 ISO 国际标准之上，针对行业应用特点，由相应的国际组织制定电信智能卡标准，USIM 卡由 3GPP 终端和 USIM 卡规范组的第三工作组 T3 制定标准，相关标准如下。

(1) 3GPP TS 31.101/ETSI TS 102221：UICC 终端接口，物理和逻辑特性。

(2) ETSI TS 102 222：电信应用的管理命令。

(3) 3GPP TS 31.102：USIM 应用特征。

(4) 3GPP TS 31.111/GSM TS 11.14/ETSI TS 102223：USIM 应用工具包。

(5) 3GPP TS 31.113/31.114：USAT 解释器字节码/USAT 解释器协议和管理。

(6) 3GPP TS 33.102：3G 安全体系结构。

(7) 3GPP TS 23.048：(U)SIM 应用工具包的安全机制 Stage 2。

(8) 3GPP TS 43.019：SIM 应用工具包 Java API。

10.5.2 USIM 卡操作系统

COS 的主要功能是控制智能卡和外界的信息交换，管理智能卡内的存储器并在卡内部完成各种命令的处理。随着智能卡性能的不断发展，卡操作系统的功能和结构也在不断发展，从最初面向专用应用的前后台结构的监控程序，发展到支持多应用的动态应用下载的 COS。

1. USIM 卡的文件系统

USIM 卡中的文件都是通过目录按照等级排列组织起来的。其中包括有一个主文件(master file，MF)。主文件下有许多不同的被称为基本文件(EF)的文件。在根目录下又有称为专用文件(DF)的子目录。每个子目录下又有基本文件。图 10-19 表示出了 UICC 中的文件结构。

2. USIM 卡指令系统

USIM 卡与手机处理芯片之间的数据交换遵循 ISO/IEC 7816-4 规定的数据格式，交换的数据信息包括各种操作命令和响应对。

图 10-20 表示出了 USIM 卡的指令执行逻辑。

3. USIM 多应用

第二代 SIM 卡仅仅是一种单应用卡，它仅遵循 GSM11.11 规范，不能直接添加额外的应用。而 USIM 卡则摆脱了这些束缚，它实现了平台和应用的分离，为此 3GPP 组织专门制定了 UICC 多应用平台规范。USIM 在 3G 通信中只是作为一个应用存在于 UICC 平台之上。其他一些非电信的应用也完全可以添加到该平台中，而且每个应用都可以遵循各自的行业规范。

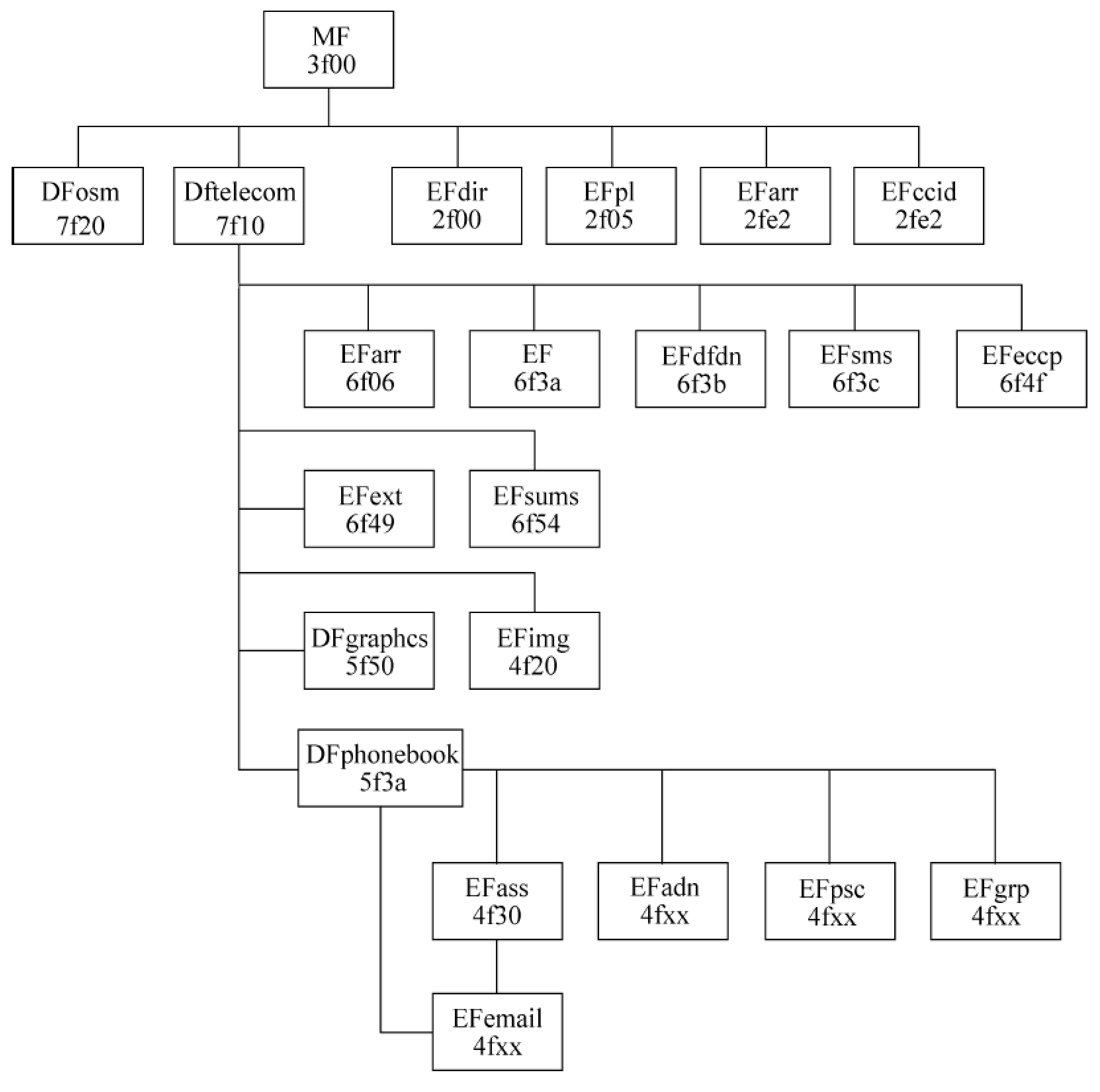


图 10-19　UICC 的文件标识符集目录结构

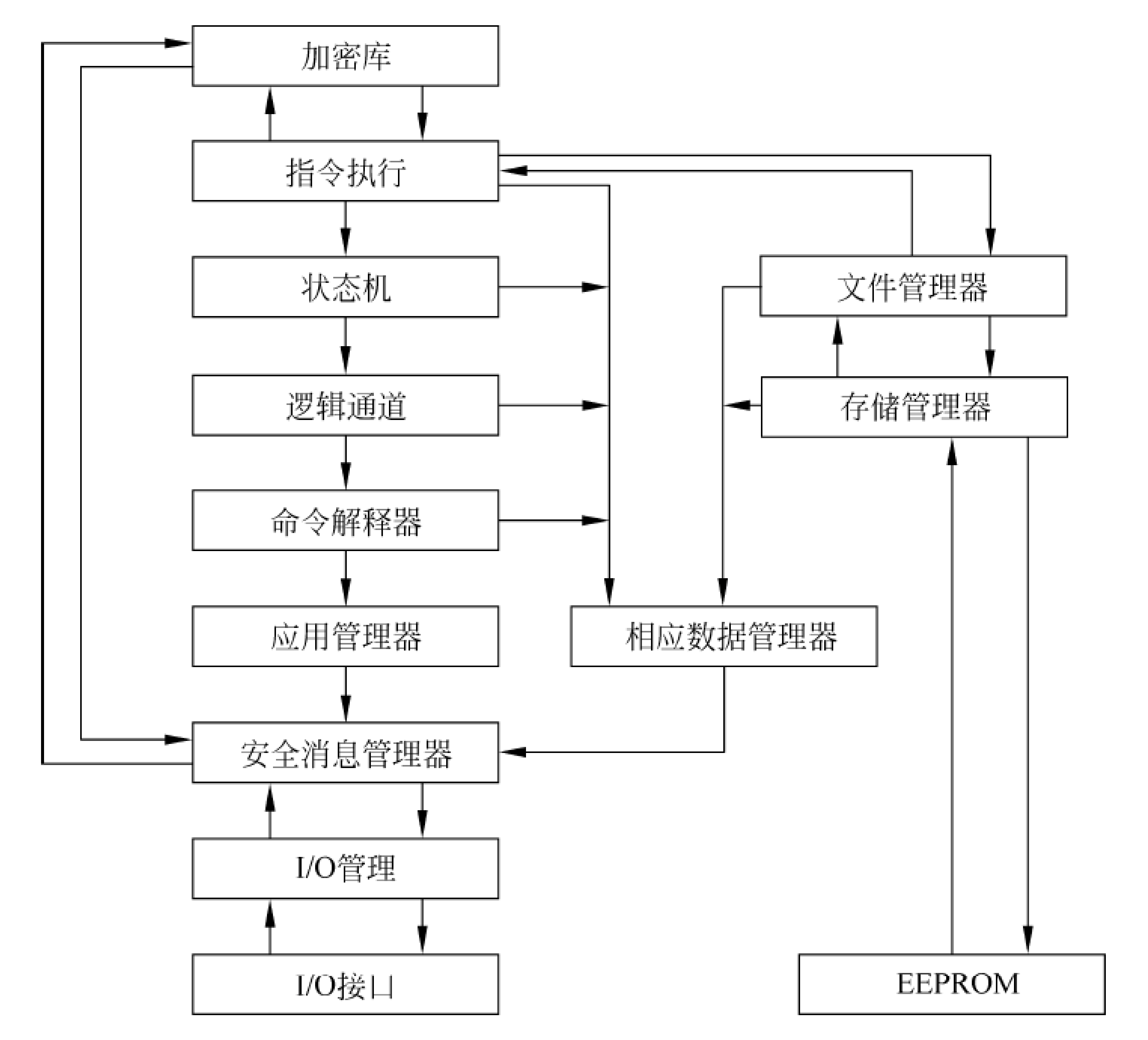


图 10-20　USIM 卡的指令执行逻辑

USIM 卡的一卡多应用不仅解决了用户在日常生活中需要保管的卡数过多的问题，更有利于规范不同行业使用相互兼容的操作系统和数据服务，杜绝了同一环境不同种类阅读机重叠所造成的浪费。在多应用中采用了内存共享方式存放共享信息，提高了系统的利用率。

USIM 卡多应用结构如图 10-21 所示。

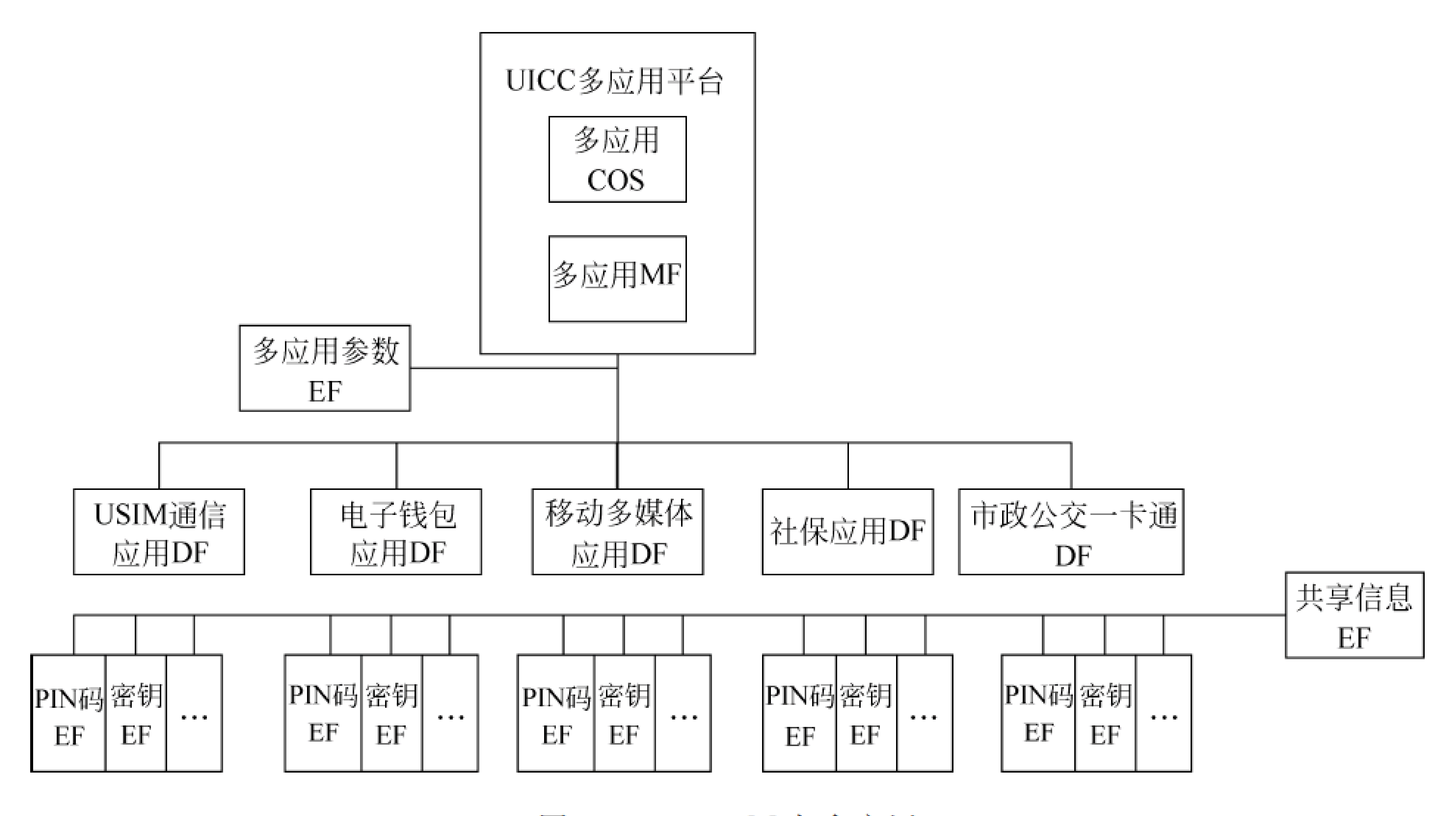


图 10-21 USIM 卡多应用

10.5.3 USIM 卡的安全体系

USIM 是 3G 网络中存储用户秘密信息和执行加密运算的模块，与鉴权和访问控制相关的安全威胁如下。

(1) 攻击者盗用终端和 UICC 卡系统资源。

(2) 攻击者在非法终端上更改 IMEI(international mobile equipment identify)访问系统资源。

(3) 破坏终端数据的完整性。

(4) 攻击者可能获得用户存储在终端或 UICC 上的个人数据，如电话本。

(5) 攻击者可能获得服务提供商存储在 UICC 上的身份认证数据，如认证密钥。

(6) 攻击者可能监听 UICC 与终端的接口，复制 USIM 卡。

清楚了智能卡可能受到的潜在威胁，下面再来讲一下智能卡的安全体系和实现方法。

智能卡的逻辑安全体系是智能卡 COS 中一个极为重要的部分，它涉及到卡的鉴别与核实方式的选择。智能卡的安全体系在概念上分为三大部分：安全状态、安全属性和安全机制。安全状态是智能卡当前或在命令操作完成后的一种工作状态。安全属性实际上是定义在执行某个命令时所需要的安全条件。安全机制可以认为是安全状态实现转移所采用的转移方法和手段，通常包括：PIN 鉴别、密码鉴别、数据鉴别以及数据加密。这就是 COS 安全体系的基本工作原理。

PIN 码一般都存在于独立的基本文件中，这样可以防止 PIN 被未授权更改。授权用户

可以通过发送 PIN 指令来完成相应的更改操作，然而对于大多数操作系统来说，多次输入 PIN 码会导致失效和被锁。

对于文件的查看和调用 COS 也有相应的安全机制，被称为文件的访问控制，国际标准 ISO/IEC 7816-8/9 已在安全属性方面作出实质性的规定。通常在内存 RAM 中设置一个 8/16 位的鉴别器，为的是对密码进行鉴别。鉴别寄存器所反映的是智能卡当前所处的安全状态。智能卡里每个文件的文件头中通常都存储该文件能够被访问的条件，一般包括读写两个条件，这就构成了该文件的安全属性。而用户可以通过向智能卡输入 PIN 就能够改变卡的安全状态。

UICC 通常采用如下 3 种安全机制。

1. 使用访问控制列表管理数据共享

UICC 使用访问控制列表(ACL)描述对文件进行各种操作的权限。每个文件都使用文件控制参数(FCP)描述文件的大小、文件名等属性以及文件的 ACL。ACL 列出的是文件的访问规则，每条访问规则都关联着由一个密钥引用组成的布尔表达式和文件操作。访问控制列表和文件有两种类型的关联方式：直接方式和引用方式。直接方式把 ACL 放入文件的 FCP 中，这种方式适用于 UICC 上文件数目较少或者每个文件 ACL 不同的情况。但实际 USIM 中采用的是引用方式，把 ACL 存放在一个专有的记录文件 EFarr 中，在文件的 FCP 中只保留 ACL 在 EFarr 文件中的记录索引。访问控制列表和文件的关联关系如图 10-22 所示。

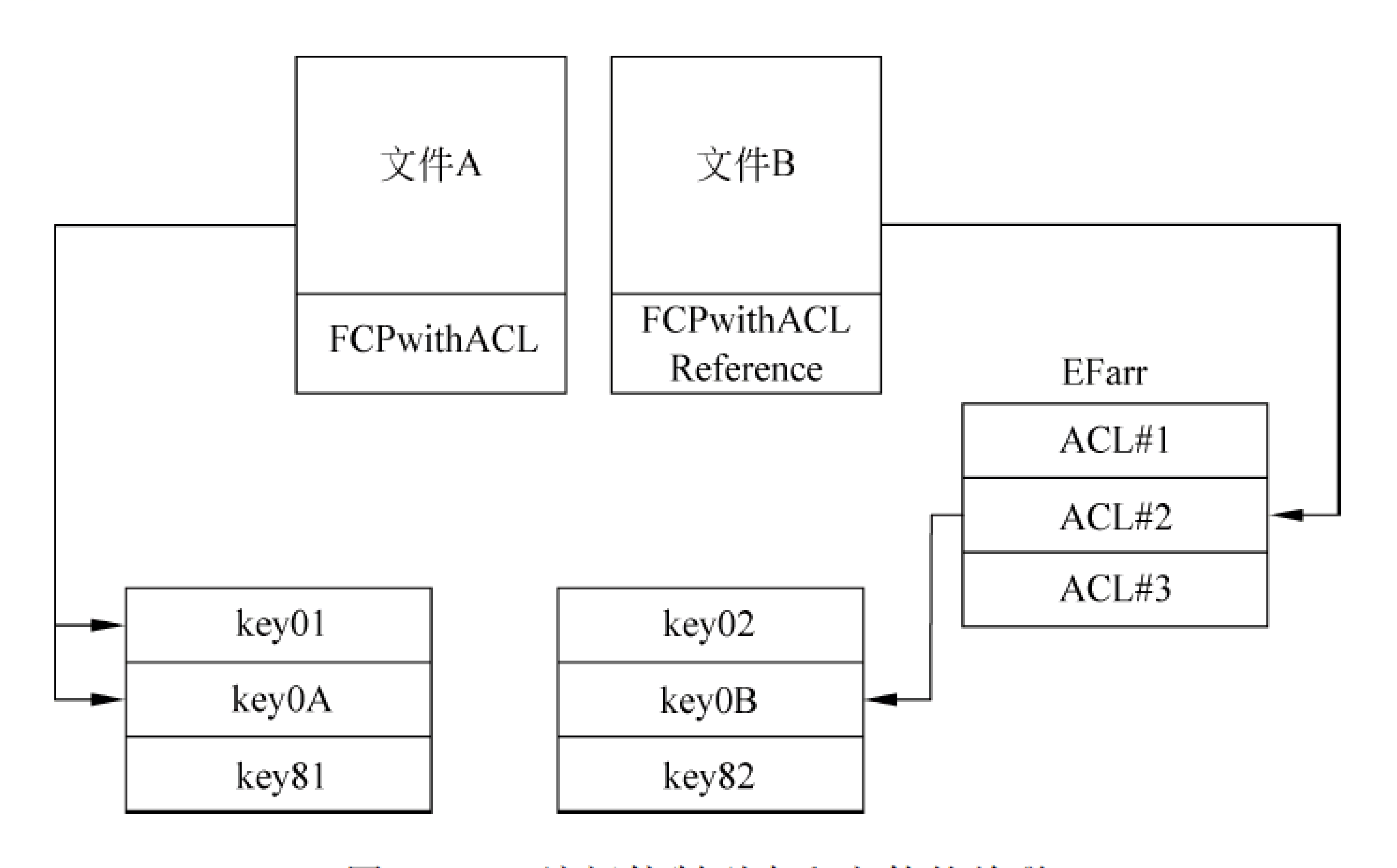


图 10-22 访问控制列表和文件的关联

USIM 卡可以为每个应用分别建立一个 EFarr 文件，也可以在整张卡上使用一个 EFarr 文件。ACL 引用采用简单的 TLV(标签、长度、值)格式存储。

2. 密钥与应用之间的关联

UICC 平台支持多应用和密钥与不同应用之间的映射关系。UICC 的访问条件共分 7 个级别：Always、PIN1、PIN2、RFU(保留)、ADM(管理 PIN)、NEWVer。其中 PIN 分为三级：通用 PIN、应用 PIN、局部 PIN。通用 PIN 是允许多个应用共享的一个全局 PIN。应用 PIN 允许具备对 UICC 上文件的全局访问权，它的校验状态可以由应用的激活或终止进程复位。

3. USIM 卡的生命周期各个阶段的安全措施

一张卡从制造出来到销毁的整个过程被称为生命周期，一般包括 5 个阶段。

（1）生产阶段。由芯片厂商完成这一阶段，为了避免伪造芯片，从这时直至芯片装入塑料卡内的期间，芯片加入一个制造密钥（KF）进行保护。每个芯片的 KF 都是由主制造密钥导出的，并且独一无二。

（2）预个人化阶段。这一阶段由卡商来实现。为了加强安全性，实现卡到发行者的安全传递，制造密钥（KF）将被个人密钥（KP）取代，然后写入个人化锁（Vper）进一步阻止 KP 被篡改。同时废除卡的物理地址存取控制指令，改用逻辑存储地址访问。这样就保护了操作系统部分及制造区域被访问和修改。

（3）个人化阶段。这一阶段由卡商来实现。建立逻辑数据结构，将数据文件内容及应用数据写入卡内。同时，将卡持有者设置的识别号、PIN 以及解锁 PIN 等信息存入卡内。

（4）使用阶段。卡持有者在这一阶段实现卡的常规使用。这时，应用系统、逻辑文件接入控制以及其他都处于激活状态。按照应用协议，卡的信息存取必须遵循安全政策。

（5）无效阶段。此状态时卡的生命将被终止。通常通过写入无效锁或锁住 PIN 的方式实现。

10.5.4 空中下载技术

过去，移动通信智能卡只是为网络提供用户身份鉴权，而且由于移动通信与智能卡技术的限制，运营商基本无法在卡端进行业务创新，所以移动智能卡的作用比较单一。近年来，随着技术的发展，围绕智能卡的一系列技术创新突破，为运营商开展丰富的卡端业务提供了可能，而 3G 时代的全面到来，移动通信智能卡朝着应用多元化的方向发展。移动通信智能卡不再只是用户身份识别的工具，它还可以为用户提供更多的服务，无时无刻不改变着人们的生活方式。下面本节将对空中下载技术进行简要介绍。

1. 空中下载技术简介

空中下载（over the air，OTA）是近年来非常流行的一种在移动通信中的个性化定制技术，通过移动通信的空中接口可以对 SIM 卡数据及应用进行远程管理，使得移动通信不但提供了移动化的语音和数据服务，而且还能够提供移动化的新业务下载。这样，应用及内容服务商可以不受平台的局限，不断开发出更具个性化的贴近用户需求的服务，如飞信、营业厅服务、互动娱乐、位置服务以及银行交易等。

典型的 OTA 系统如图 10-23 所示，这是一个空中接口采用短消息技术的 OTA 系统。用户首先请求下载服务，可以通过两种方式。一种是通过手机发送短消息，短信通过短信中心投递到短消息网关，短消息网关起到消息解释的作用，随后将解释后的信息传递给 OTA 服务器。另一种是通过网上定制的方式，用户在因特网上注册使得网上的用户账号和用户手机号码进行绑定，当用户登录网上系统，可根据网上提供的信息向 OTA 服务器提交下载申请。OTA 服务器收到下载申请后，将应用或数据发送给短消息网关，网关将应用或数据打包封装成短消息的格式发送给短消息中心，最后转发到用户。

空中接口不但可以采用短消息技术，还可以采用 WAP、GPRS 和 CDMA1X 等。目前国内 OTA 系统空中接口大多采用短消息技术。因此受限于短信的局限性，每条短信的长度有限，且受制于短信中心的处理能力，只适于下载一些数据量比较小的应用。而且短信通

道是面向无连接服务的,数据可能在传输过程中丢失,因此不适用下载高安全性的应用或数据。GPRS 的传输速率要远高于短消息,且采用面向连接的通信方式,可以对丢失、损坏的数据包进行重发,保证了数据的完整性。所以从理论上说,采用 GPRS 作为空中接口要比采用短消息更加安全,而且速率更快。目前 OTA 业务的普及度并不十分高,业务数量也比较有限,随着 3G 的到来,数据传输速度大大提升,我们期待着更加丰富的 OTA 业务,帮助我们随时获得最新的资源。因此采用传统的短消息技术作为空中接口可能会渐渐跟不上需求的发展,采用高速率、高安全性的 GPRS 空中接口才是未来的发展趋势。

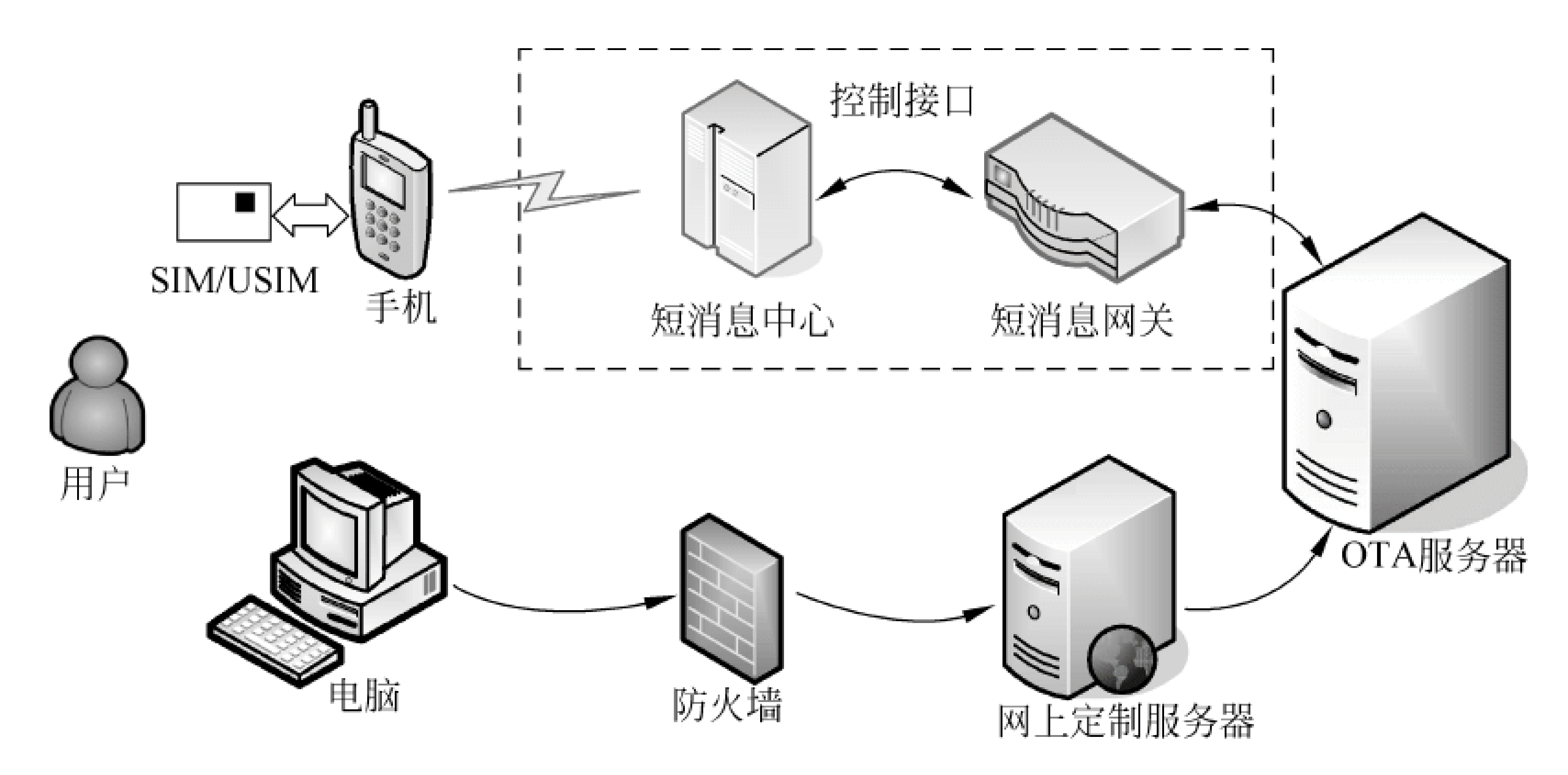


图 10-23　OTA 系统示意图

2. 空中下载安全机制

OTA 技术要想大规模普及,需要有成熟、高效的安全机制。下面针对 OTA 系统中的 SIM/USIM 卡的安全机制做一下简要介绍。

传统的 2G(如 GSM、CDMA)是采用 SIM 卡。具有 OTA 功能的 SIM 卡主要拥有以下几种安全特性:

(1) 用户认证。也称网络鉴权,是网络对 SIM 卡身份的合法性的认证。

(2) 终端认证。是 SIM 卡对手机用户身份的认证,防止合法的 SIM 卡被其他非法的人盗用,插在非法的手机上。一般是通过 PIN 认证完成对手机用户身份的鉴别。每次开机需要输入合法的 PIN 码才能使用 SIM 卡功能。或者 SIM 卡记忆当前合法的手机,当 SIM 侦测到插入其他手机时,需要校验 PIN 码才能正常使用。

(3) 与 OTA 服务器的双向认证。OTA 下载过程需要 4 个认证,分别是 OTA 服务器对 SIM 卡的身份认证、SIM 卡对 OTA 下载服务器的身份认证、OTA 下载服务器对请求数据合法性的认证和 SIM 卡对菜单数据合法性的认证。双向认证可以通过基于对称密码体制的挑战应答方式、MAC 校验方式、WPKI 数字证书等方式实现。

(4) 文件与密钥的安全保护。SIM 卡中有许多重要的信息,如国际移动用户号(IMSI)、鉴权密钥等,一旦被非法访问,用户的合法权益很有可能遭受侵犯(如复制卡片、非法修改卡信息等)。所以建立一套健全的文件系统,设置严格的文件访问控制方式是非常重要的。

(5) 数据备份与恢复(防插拔保护)。在下载过程中,为防止发生任何意外情况,如掉电、插拔等而导致卡内信息处于一种未知状态,SIM 卡需要在意外情况发生后,将卡内的重

要数据恢复到初始状态。

（6）OTA 下行数据的安全处理。主要是鉴别数据的完整性和有效性，并对数据正确解析。完整性鉴别可以通过附加 MAC 实现，防止数据被修改或替代。有效性可以采用同步计数器机制，这样就可以判断数据是新鲜的，不是因网络堵塞或重放攻击而导致消息的延迟。在鉴别完成后，需要将数据解封装，例如判断当前数据是明文或密文、对数据结构正确解析等。

（7）与服务器的同步。为了使 OTA 服务器更好地管理下载业务和用户信息数据库，需要将用户端 SIM 卡已下载的业务信息与服务器端的信息保持同步。SIM 卡和服务器都可以主动要求更新同步。

3. 3G 时代的 OTA 技术

3G 采用的是 USIM 卡，最大的变化是支持一卡多用，这样极大地丰富 OTA 业务，使它不局限在电信方面的应用，在交通业、零售业等领域也会有广泛的应用。所以说 3G 的到来推动了 OTA 技术的发展。具有 OTA 功能的 USIM 卡又增加了以下几种安全特性。

（1）用户的双向认证。2G 的 SIM 卡的用户认证是网络对 SIM 卡的单向认证，3G 的 USIM 卡的用户认证是网络与 USIM 卡之间的双向认证，即保证了用户身份的合法性，又保证了网络的合法性。

（2）一卡多应用的相关安全机制。USIM 卡的一卡多应用带来了一些新的安全特性，例如为不同的应用设置不同级别的 PIN 管理不同的应用、多逻辑通道的安全管理、应用间的防火墙、数据的隔离和共享等。

（3）高速下载通道。USIM 卡引入了全新的 BIP（bearer independent protocol）协议接口，通过 BIP 协议结合 USAT 应用，手机终端允许 USIM 卡和远程服务器之间进行透明的数据传输。BIP 协议更有利于实现高速移动数据业务的传输，使得各种业务数据下载变得更加容易、快捷。

（4）补丁下载机制。程序补丁下载机制可以对卡片操作系统以及程序进行升级和错误修正，包括对卡片操作系统进行升级和错误修正、API 函数的修改和扩充、上下行指令的修改和扩充、支持的安全算法的扩展，例如增加新的密钥分散算法或者加解密算法等。补丁下载的管理可以通过动态链接库进行，动态链接库通过定义一系列 API 实现了对 USIM 卡的补丁下载进行控制和管理。补丁下载先要进行服务器与动态链接库的双向认证，保证了双方身份的真实性，然后再进行补丁下载的流程。

OTA 技术是一个系统级工程，上面只是对 SIM/USIM 卡的安全特性进行了简要介绍。在 OTA 系统中，还有 OTA 服务器的安全设计、通信链路的安全协议等许多关键安全技术，由于篇幅有限在此不再过多介绍，有兴趣的读者可以查阅相关资料。在不远的将来，OTA 技术将呈现出下载多样性、系统通用性、业务灵活性、推广快速性的特点。而作为 OTA 技术载体的 SIM 技术也在进行着不断的更新换代。兆级图像化 SIM 卡、USIM 的出现为承载更多、更复杂的 OTA 应用提供了技术支撑。随着 OTA 技术的不断发展和 SIM 的升级换代，未来基于 OTA 技术的 SIM 应用将变得更加丰富多彩，成为移动运营商市场竞争的重要资源，有着令人期待的光明前景。

10.5.5 手机钱包

近年来，随着手机的普及和人们消费能力的不断提升，移动电子商务应运而生。虽然移

动支付的深层次应用还处于市场培育期，但随着一些试验性服务和产品的推出，移动支付越来越被人们所接受，拥有了广泛的用户基础。

如图 10-24 所示，移动支付技术（本节主要讲手机支付技术）的实现方式主要分为近距离支付和远距离支付两种。

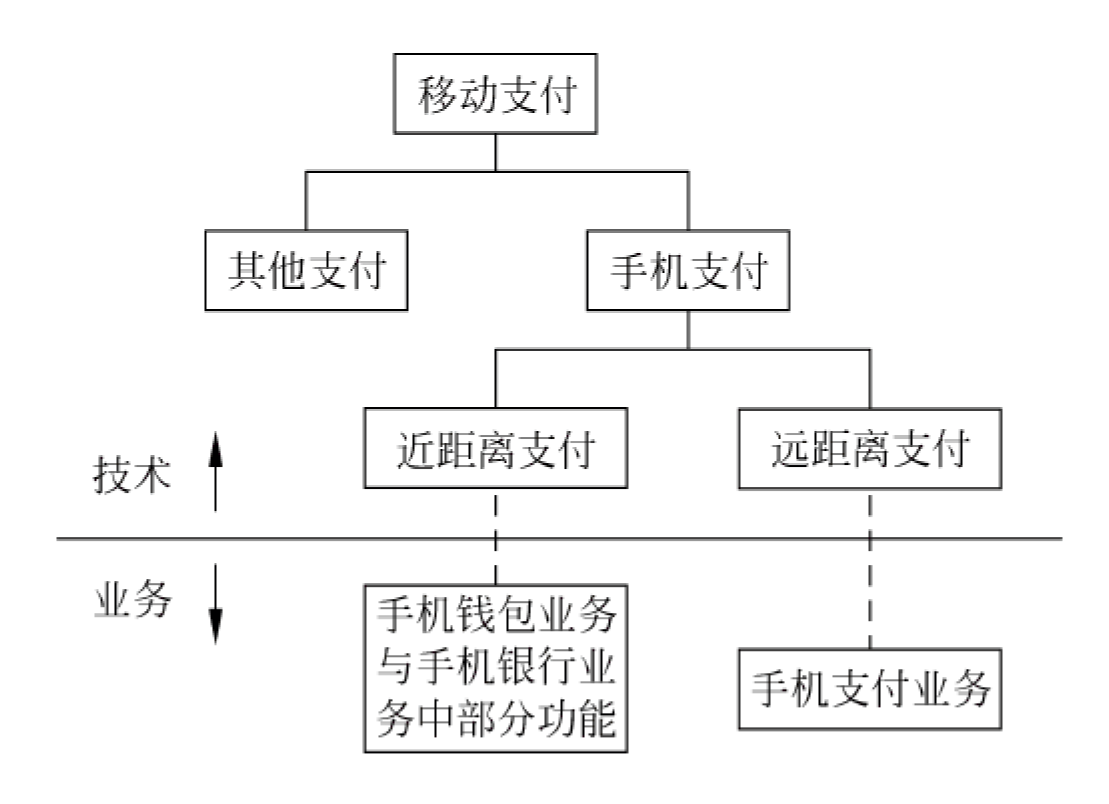


图 10-24　移动支付技术与业务对应图

远距离支付是通过短信、WAP 等远程控制完成支付，目前推出的手机支付业务就是利用此项技术实现的。资料显示，从运营商初期探索的教训看来，远程支付并不为市场接受，而非接触式近距离移动支付业务拥有现场支付、实时支付的特点，更容易被人们接受。近距离支付大多是利用 RFID 技术实现的，也有利用红外、蓝牙等技术实现的，但从运营成本和适用性综合考虑，利用 RFID 是最易被人接受的，目前推出的手机钱包业务就是利用此项技术实现的。

手机钱包作为移动支付的一个应用分支，结合了电子化货币、身份验证、移动通信与移动终端的崭新业务，使用户可以随时随地享受多种服务，如移动支付、身份验证、防伪、广告、信息交换等服务。手机钱包（对应近距离移动支付技术）的主流解决方案目前主要有 3 种：SIMPASS 方案、RF-SIM 方案和 NFC 方案。

1. SIMPASS 方案

SIMPASS 方案实质上就是利用双界面智能卡实现近距离移动支付的技术。SIMPASS 支持传统的 7816 通信接口（与手机相连）与非接触 14443 通信接口（完成支付）。14443 接口需要增加天线，一种方法是将天线集成在 SIM 卡里面，在 SIM 卡上拖一条辫子；另一种方法是定制手机，将天线固定在电池上，需要占用 SIM 卡两个不同的触点，如图 10-25 所示。SIMPASS 方案的优点是价格低廉，不需要更换手机，只需换一张双界面 SIM 卡并增加一根天线，而且采用非常成熟的 ISO/IEC 14443 标准，使用和迁移很简单。缺点是占用了 SIM 卡的两个为将来高速 USB 接口使用的触点，对手机机型有一定限制（不适用于手机后

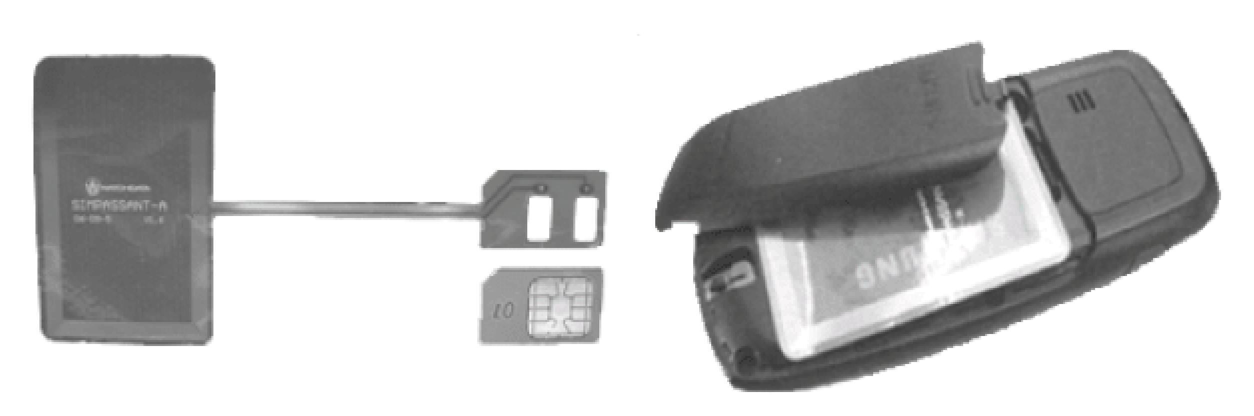


图 10-25　SIMPASS 方案实物图

盖与电池一体的情况)，采用外置天线方式，线圈易损坏，产品易用性、可靠性不足。目前握奇数据公司主推的 SIMPASS 技术方案，在江苏、广东等地已发展大批用户。

2. RF-SIM 方案

RF-SIM 方案是将射频模块和天线集成到 SIM 卡中，使用 2.4G 频率的无线接口进行近/中距离无线通信的技术。RF-SIM 技术变无源为有源，最大通信距离可达 15m，可根据用户需要进行调整。RF-SIM 方案的优点是不需更换手机，同时对手机类型没有限制，换一张 RF-SIM 卡即可，如图 10-26 所示。缺点是 2.4GHz 频段支付应用没有相关国际标准，也没有系统的安全性检测，支持的厂家比较少，而且与目前广泛使用的 13.56MHz 的阅读器不兼容，需要对这些阅读器进行升级，成本比较高。中国移动在 2009 年 11 月向北京、内蒙古、吉林、上海、浙江、福建、湖北、湖南、广东、重庆等地全面开启采用 RF-SIM 方案的手机钱包业务。

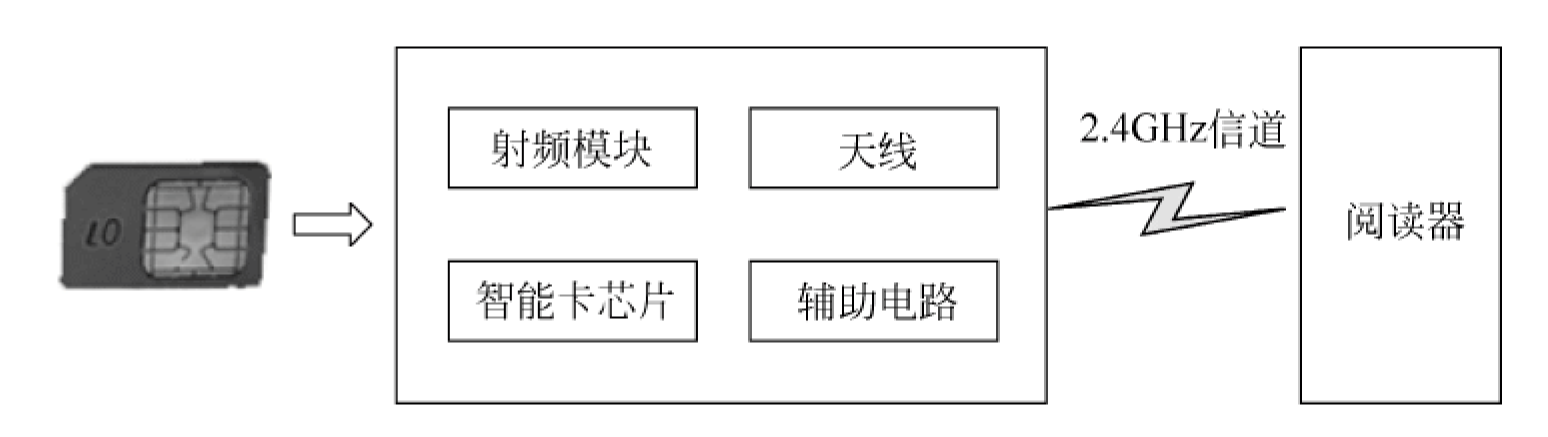


图 10-26　RF-SIM 方案示意图

3. NFC 方案

NFC(near field communication)是由飞利浦公司联合诺基亚、索尼等著名厂商主推的一项近距离无线通信技术。NFC 方案可以将各种卡(银行卡、公交卡等)通过 OTA 下载到 NFC 移动终端的 SIM/USIM 卡上进行移动支付。NFC 方案由近场通信控制芯片和 SIM 卡组成，近场通信控制芯片完成射频信号(13.56MHz 的频段)的转换和识别；SIM 卡上安装有用户的应用，与近场通信控制芯片之间采用单线通信协议实现数据传输，如图 10-27 所示。

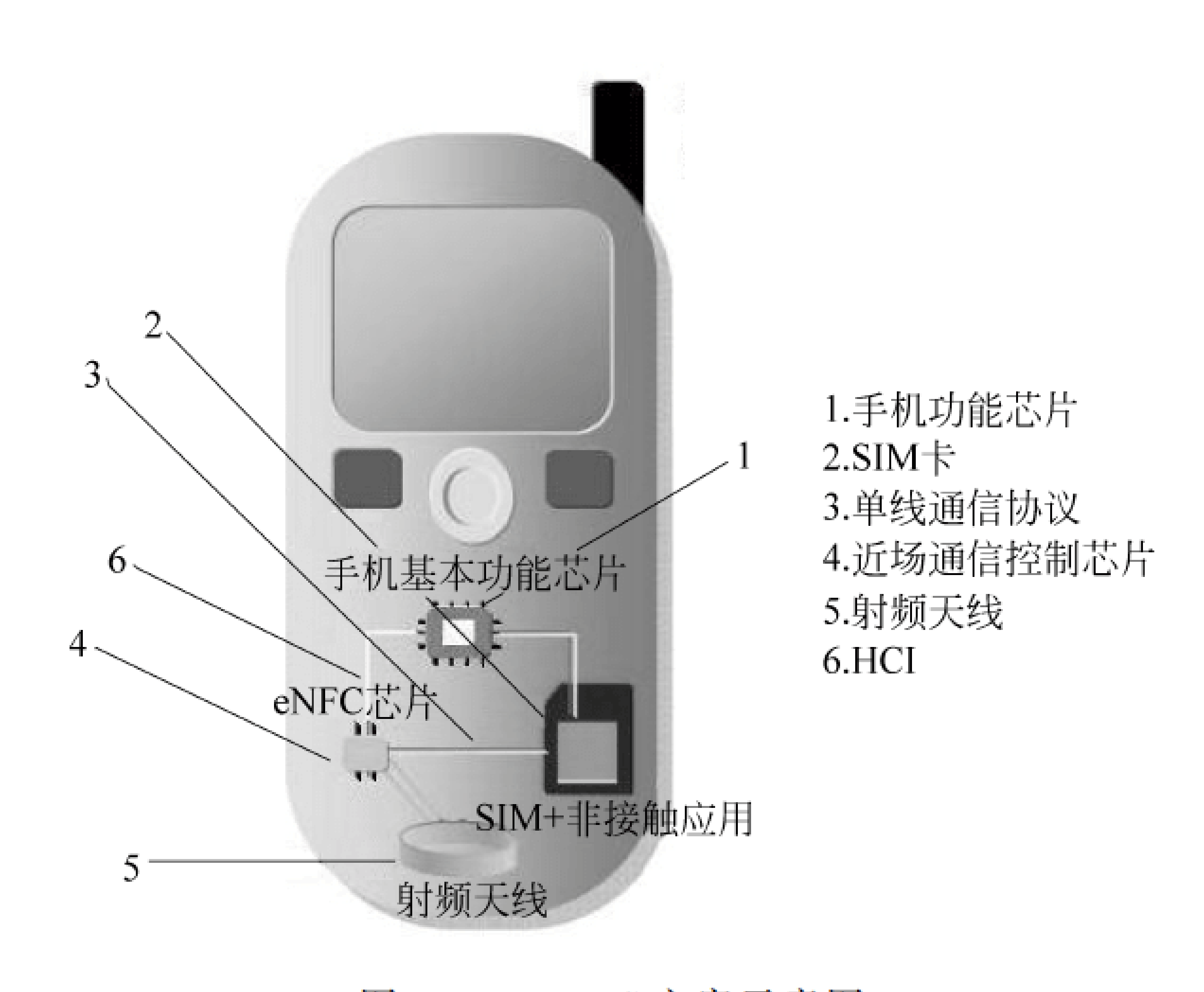


图 10-27　NFC 方案示意图

NFC 方案拥有以下 3 种通信模式：

(1) 主动模式。NFC 终端作为一个读卡器，主动发出自己的射频场去识别和读写别的 NFC 设备。

(2) 被动模式。NFC 终端可以模拟成一个卡被读写，它只在其他设备发出的射频场中被动响应。

(3) 双向模式。双方都主动发出射频场来建立点对点的通信。

NFC 方案的优点是拥有多种通信模式，NFC 手机不仅能当智能卡，也可以当阅读器使用，还可以与其他 NFC 手机点对点传输数据，和前两种方案相比，无论从技术上还是从业务应用上可扩展性都很强。缺点是需要使用具有 NFC 功能的手机，更新成本很高。NFC 技术是国际上最被认可的开放式商用和技术框架，在北美和欧洲应用比较广泛。

这 3 种解决方案都具有加密功能，可满足当前手机钱包小额支付的安全需要。RF-SIM 采用的是扩频通信，将传送的信息扩展到很宽的频带上去，其功率密度随频谱的展宽而降低，甚至可以将信号淹没在噪声中，因此其在无线信道上的保密性很强；而 NFC 拥有强大的技术阵营支持，有提供更高级的安全防护的潜力，未来应用空间可能会更广阔，所以从安全角度分析，RF-SIM 和 NFC 各有千秋。

随着 3G 的到来，全新的应用将渐渐吸引人们的视线，相信手机钱包这个业务会迎合相当一部分人群的胃口，尤其是喜爱尝试新元素的年轻人和玩机一族。站在技术开发的角度，移动支付的安全性是否能抵御越发高科技的破解技术，这是一个十分令人关注的课题。站在运营商的角度，手机银行、手机支付、手机钱包这些移动支付业务，与市政公交一卡通，和传统的信用卡付费如何争夺支付市场，这将是一场精彩的博弈。站在用户的角度，面对当前种类繁多的支付手段，用户心中的天平将会向安全性更让人放心、应用范围更广、增值业务更丰富的支付手段倾斜，技术开发商和运营商将会绞尽脑汁，用更完善的技术和更优质的服务抓住更多用户的心。在未来支付观念的变革浪潮中，手机钱包将何去何从，让我们拭目以待！

参考文献

[1] 高雪峰，徐亦书. 论城市公共交通一卡通系统应用发展趋势. 金卡工程，2003，7(4)：60～65

[2] 徐明. 上海市公共交通一卡通系统的建设与发展. 建设科技，2002，(12)：25～28

[3] 刘小明，荣建，关宏志. 北京交通现状与智能交通系统的发展. 交通运输工程与信息学报，2003，1(1)：48～56

[4] 谢希仁. 计算机网络. 第 4 版. 北京：电子工业出版社，2003

[5] 温利新. 城市一卡通—公用事业智能卡应用系统解决方案. 金卡工程，2003，(3)：8～13

[6] 郑莉. 公共事务及市民一卡通系统的网络设计. 计算机工程与设计，2005，26(8)：2240～2242

[7] 关振胜. 公钥基础设施 PKI 及其应用. 北京：电子工业出版社，2008

[8] 孙森. 网络银行. 北京：中国金融出版社，2004

[9] 中国金融认证中心. 网上银行安全性成关注焦点——2005CFCA 网上银行调查报告. 中国计算机用户，2005，(49)

[10] 段泽强. 中国网上银行面临三大问题. 金卡工程，2005，9(03)：52～55

[11] 徐志大，南相浩. Internet X. 509PKI 安全通信协议设计与证明. 计算机工程与应用，2003，39(1)：161～164

[12] 郭伟乔,荣川.数字时间戳协议 RFC3161 设计与实现.宇航计测技术,2006,26(5):41～44
[13] 刘剑峰,肖文康.RFID 和 EPC 基本工作原理与应用介绍.条码与信息系统,2006,(5):29～31
[14] 姜琼惠.基于 B/S 模式的物流管理系统设计与实现.长沙:中南大学,2007
[15] 王岭,海刚.浅谈 RFID 技术在供应链管理中的应用价值.技术经济与管理研究,2006(6):30～32
[16] 马庆容,俞军,张纲.国内 RFID 应用及产业研究.中国集成电路,2006,(11):11～15
[17] 元媛,姜岩峰.射频识别(RFID)技术综述.半导体技术,2006,(11):3～4
[18] 孙剑斌.基于 RFID 技术的物流管理系统中国管理信息化.中国管理信息化,2006,9(5):34～35
[19] 池亚萍,王全民.基于 USBKey 的可信平台模块的研究与仿真设计.北京电子科技学院学报,2007,15(4):13～15
[20] 王震宇,刘鑫杰,任杰等.嵌入式终端可信计算环境的关键技术.计算机工程,2008,34(33):239～241
[21] 李勇,王飞,刘威鹏.基于可信计算平台的用户身份认证研究.信息安全,2009,25(24):41～43
[22] 彭彦,鞠磊,方勇.基于 Java 智能卡的可信度量模块设计.微计算机信息,2009,25(18):40～42
[23] 李振华.基于 USBKey 的 EFI 可信引导的设计与实现.北京:北京交通大学,2007
[24] 许天亮,方勇.基于可信计算的 Java 智能卡的设计与实现.嵌入式系统应用,2009,25(17):45～47
[25] 张鲁国,李峥.基于可信计算的智能卡安全保护框架设计.信息安全与通信保密,2009,(08):126～129
[26] 许天亮.基于 Java 智能卡的可信认证技术的研究与实现.西安:西安电子大学,2009
[27] Challener D,赵波等译.可信计算.北京:机械工业出版社,2009
[28] 全卫新.基于 3G+USIM 卡的空中下载技术研究与实现.上海:上海大学,2008
[29] 张喜蕊.基于 SIM/USIM 卡的 OTA 技术研究.北京:北京邮电大学,2008
[30] 朱正键,达飞鹏,阙朝阳等.基于 OTA 技术的 SIM 应用研究.通信市场,2009,(03):100～108
[31] 陈欣.移动支付技术比拼 谁能笑到最后.卡市场,2009,(09):25～27
[32] 王淑君,吴军.射频技术与手机结合模式浅析.硅谷,2009,(05):24～25



系统测试

使用手工或者自动手段来运行或者测定某个系统的过程，其目的在于检验它是否满足规定的需求或者是弄清预期结果与实际结果之间的差别，这就是测试的定义。

测试工作在智能卡应用系统开发工作中占有重要地位，尤其对于位于最底层的装载有嵌入式COS系统的智能卡。智能卡在通用性、可靠性、可用性、安全性等方面有着严格的要求，必须对其进行严格的测试，以符合国际标准、国内标准或者行业标准。测试的完备性标志着智能卡是否还存在可能被攻击的安全漏洞，无论硬件或者软件、无论流程还是算法，只要不符合标准或不通过综合测试，都可能成为攻击者殷切盼望的攻击对象。

本章按照测试层次的顺序依次阐述了物理测试、协议测试、COS测试和业务测试，并将通用性、可靠性、可用性和安全性测试穿插其中。

11.1 测试层次

依照因特网的OSI参考模型，可以将智能卡的协议划分为七层，如图11-1所示，而测试工作也同样是根据这七层进行展开。

第七层：应用数据
第六层：应用协议
第五层：未应用
第四层：传输协议
第三层：协议激活
第二层：能量和信号接口
第一层：物理特性

图11-1 智能卡“传输”协议的七层模型

第一层物理层对智能卡的物理特性和尺寸等进行了要求，对智能卡在紫外线、X射线、交流电场、交流磁场、静电、静磁场、工作温度、动态弯曲和动态扭曲等方面提出了要求。

第二层能量和信号接口层描述了智能卡的能量传送、信号接口、电气特性等特性。

第三层协议激活层，描述了智能卡的初始化和抗碰撞流程。

第四层传输协议层包括接触式卡的面向字节的传输协议T=0和面向分组的传输协议T=1，非接触式卡的传输协议T=CL。

第五层对应OSI模型中的会话层，在智能卡通信协议中未采用。

第六层应用协议层，对具体应用中的协议进行了描述，如在ISO/IEC 7816-4中规定的应用命令。

第七层应用数据层，对具体应用中数据的逻辑数据存储结构进行了描述。

智能卡的测试可依据智能卡通信协议的七层模型进行测试，图11-2所示为智能卡通信协议七层模型与参考标准之间的关系。

将图11-2中所示的七层模型可进一步的归类，可将第一层的物理特性和第二层的能量和信号接口归类为物理和硬件测试；将第三层协议激活和第四层传输协议归类为对协议的

测试；将第六层应用协议和第七层应用数据归类为应用业务测试。本章将从这三个层次展开，对智能卡测试进行系统的介绍。由于 COS 在智能卡应用中有着举足轻重的作用，单列一节对 COS 的测试方法进行介绍。

层次	接触式	非接触式	
第七层	ICAO DOC 9303 EMV 2000 GSM 11.11…	ICAO DOC 9303 EMV 2000 GSM 11.11…	应用业务测试
第六层	7816-4	7816-4	
第四层	10373-3 7816-3	14443-4	协议测试
第三层	10373-3 7816-3	14443-4	
第二层	10373-3 7816-2 7816-3	10373-10373-7 14443-2	物理和硬件测试
第一层	10373-1 7816-1	10373-1 14443-1	

图 11-2 智能卡七层通信模型与参考标准之间的关系

目前，很多国际组织都推出了针对智能卡的测试标准：如国际标准化组织（ISO）制定了针对识别卡的测试标准 ISO/IEC 10373；如国际民航组织（ICAO）提出了针对电子护照的测试标准；如 EMV 组织也提出了针对借记卡和信用卡的测试标准；如欧洲电信标准委员会（ETSI）提出了针对 SIM 卡的行业规范 GSM 11.11。这些规范在测试过程中都要全面考虑，并实施在测试工作中。

11.2 物理和硬件测试

物理特性是智能卡的硬件基础。物理和硬件测试内容可分为对一般特性的测试，以及物理和电气特性的测试。一般特性的测试包括卡的使用温度、机械强度、紫外线防护能力、电磁场影响等参数的测试；物理和电气特性包括对卡的触点电阻和阻抗、静电、点压力测试、VCC 电流等参数的测试。另外真随机数发生器作为保证智能卡安全的重要设备，本节的最后一部分对随机数的随机性测试进行了介绍。

11.2.1 一般特性测试

ISO/IEC 10373 标准由七部分组成，如表 11-1 所示。

表 11-1 ISO/IEC 10373 的组成

序　　号	内　　容
第一部分	一般特性测试
第二部分	磁条卡测试
第三部分	接触式集成电路卡和相关接口设备测试
第四部分	密接触卡测试
第五部分	光存储卡测试
第六部分	近接触卡测试
第七部分	疏接触卡测试

其中，ISO/IEC 10373-1 对智能卡一般特性的测试方法进行了规定，测试方法如表 11-2 所示。

表 11-2　ISO/IEC 10373-1 中规定的 16 种测试方法

测 试 方 法	测 试 目 的
1 卡翘曲	测试卡片能够翘曲的程度
2 卡尺寸	测试卡片的高度、宽度和厚度
3 拉伸强度	测试卡片各层之间的拉伸强度
4 耐化学性	测试各种化学污染对被测卡片的有害影响
5 在给定温度和湿度情况下，卡的尺寸和翘曲的稳定性	测试将待测卡片暴露在指定的温度和湿度环境中以后，卡片的尺寸和平整度是否能够符合标准中的要求
6 卡的粘连	测试将多张被测卡片堆在一起可能带来的不利影响
7 弯曲硬度	测试卡片的弯曲硬度是否在标准规定的范围内
8 动态弯曲压力	测试对卡片施加弯曲力以后可能对机械和功能上的影响
9 动态扭曲压力	测试对卡片多次扭曲以后可能对机械和电气的影响
10 阻光度	测试样品中规定区域的阻光度
11 紫外线	测试卡片暴露在紫外线下而引起的不良反应
12 X-射线	测试卡片暴露在 X 射线下而引起的不良反应
13 静电磁场	测试卡片暴露在静电磁场中而引起的不良反应
14 字符的压花浮雕高度	测试压花卡浮雕字符的高度
15 耐热性	测试将卡片暴露在一定的温度之中以后，是否能够保持卡片结构的稳定，如图 11-3 所示
16 表面扭曲和突出区域	该测试与 ISO/IEC 10373-2 中对磁条卡的高度和表面外观测试的仪器和流程相同

该标准一共规定了 16 种测试方法，在该标准中对每项测试的目的、过程、仪器、测试报告和具体参数等有详细的描述，详情请参考该标准。



图 11-3　高低温物理测试箱

11.2.2 物理和电气特性测试

智能卡从传输协议上可分为接触式和非接触式两种类型，测试也必须分开进行。ISO/IEC 7816-2 和 ISO/IEC 7816-3 对接触式卡的物理和电气特性进行了说明，内容包括触点位置、触点阻抗、信号频率、电压值、电流值等物理特性。ISO/IEC 14443-2 对非接触式卡的能量和信号接口进行了规定。下面进行具体介绍。

（1）接触式智能卡的物理和电气特性测试

ISO/IEC 10373-3 对接触式智能卡的物理和电气特性测试方法进行了介绍，如表 11-3 所示。

表 11-3 接触式智能卡的物理和电气特性测试

测 试 项 目	说　　明
1 触点位置	测量卡上各触点的位置是否符合 ISO/IEC 7816-2 的标准
2 静电	测试静电电位对 IC 卡的影响
3 触点的电阻和电抗	使用测试卡连接器来测试卡触点表面和卡连接器之间的电阻和电抗
4 触点表面轮廓	测量被测卡片每个触点的表面轮廓及其周围的卡表面轮廓
5 触点高度	测量被测卡片的模块相对于卡塑料表面的最大高度
6 机械强度	根据应用要求对卡的机械强度进行分类
7 点压力测试	测试卡上某一点能承受集中压力的能力
8 点畸变测试	测试卡上模块区域中由弯曲压力而产生的不利影响
9 模块附着力的拉力测试	使模块经受一个试图将其从卡基片上分开的力，以确定模块是否足以附着在卡上
10 I/O 触点	在正常操作条件下，测量 I/O 触点在发送期间的输出电压和在接收方式期间的输入电流
11 CLK 触点	测量卡的 CLK 触点上电流
12 RST 触点	测量卡的 RST 触点的电流
13 VCC 电流	测量卡的 VCC 触点的电流

（2）非接触式智能卡的物理和电气测试

ISO/IEC 10373-6 对非接触式智能卡的测试方法进行了规范，分为非接触式智能卡的静电测试和功能测试。功能测试主要包括测试仪器和测试电路，PICC 和 PCD 的测试方法。同时 ISO 也推出了 ISO/IEC 10373-6 的 5 个增补，对非接触式智能卡测试进行了更全面的介绍。表 11-4 所示为 ISO/IEC 10373-6 及其 5 个增补版本的说明。

表 11-4 ISO/IEC 10373-6 及其增补

标　　准	说　　明
ISO/IEC 10373-6	近接触卡测试方法，主要包括静电测试和功能测试
ISO/IEC 10373-6 增补 1	近接触卡协议测试
ISO/IEC 10373-6 增补 2	增强 RF 测试
ISO/IEC 10373-6 增补 3	近耦合设备协议测试
ISO/IEC 10373-6 增补 4	PCD RF 接口和 PICC 可变场的增补测试
ISO/IEC 10373-6 增补 5	fc/64，fc/32 和 fc/16 的码率

ISO/IEC 10373-6 和增补 2 对智能卡物理和电器特性的测试方法进行了介绍，包括智能卡静电测试、测试仪器和测试电路的搭建，智能卡负载调制信号的幅度测试等内容。

11.2.3 真随机数测试

随机数技术是智能卡芯片安全技术中的源头。芯片的认证体系、加密算法的密钥产生、各类加解密和签名应用协议、各类抗攻击的安全措施往往都是从利用随机数开始的。因此，在智能卡芯片中如何实现高质量的随机数发生器是一项至关重要的安全技术，它所产生的随机数的随机性在一定程度上决定了整个智能卡的安全。

随机数分为真随机数和伪随机数。在智能卡的高级芯片中随机数基本上都是真随机数，而比较低级的芯片中有的就采用的是伪随机数。

真随机数由物理方法产生，取自大自然的随机现象（如宇宙噪声、电路的热噪声和放射性衰变），即使有关产生随机数方法的所有信息都被暴露，也无法预测其结果，即高质量的真随机数的周期为无穷大并且具有不可预测性。产生真随机数的常见方法有以下 3 种。

(1) 电路噪声放大法。芯片电路中的噪声主要来源有散粒噪声、接触噪声、突发噪声、雪崩噪声、热噪声等。在芯片设计中，电路中大电阻的热噪声是最易于获得的随机物理信号。一般都采用噪声迭代环原理来提高输出数据的不可预测性，使得几次迭代后的电路的状态完全无法估计，更无从预测。

(2) 振荡采样法。即通过一个高电平触发的 D 触发器把两个独立的方波进行数字混合，用低速波来采样高速波，这种方法是利用环形振荡器的频率抖动作为随机源产生随机数的。环形振荡器产生的低频时钟作为 D 触发器的时钟输入端。压控振荡器产生的高频数据作为 D 触发器的数据输入端。经过 D 触发器采样输出后，产生一位真随机数。输出端经过伪随机网络后，通过 D/A 转换电路，反馈到压控振荡器的输入。

(3) 混沌电路。利用混沌电路的不可预测以及对初始条件敏感的依赖性等本质特点产生随机数。

伪随机数用数学的方法产生，它在一些统计特性方面接近随机序列，但它是周期的和可预测的。

常见的伪随机数产生方法如 n 级 LFSR、线性同余法（linear congruence generator，LCG），和循环加密法等，它们的实现如下：

(1) n 级线性反馈移位寄存器（LFSR）。初态 n 可看作是种子，但它至多可再产生 2^n-n-1 位伪随机数。LFSR 生成随机数具有快速的特点，但其安全性较低。

(2) 线性同余法。选取足够大的正整数 M 和任意自然数 n_0，a，b，由递推公式

$$n_{i+1} = (a * f(n_i) + b) \bmod M \quad i = 0,1,\cdots,M-1$$

生成数值序列。该序列被称为是同余序列。当函数 $f(n)$ 为线性函数时，即得到线性同余序列

$$n_{i+1} = (a * n_i + b) \bmod M \quad i = 0,1,\cdots,M-1$$

由于线性同余算法是一种确定性算法，当参数 a，b，M 给定后，唯一可变的是初值 n_0。因此，一旦 n_0 选定，它将依次产生一个确定的随机数序列。在算法已知的情况下，攻击者可以通过截获 3 个连续随机数，采用解方程的方法得到参数 a、b、M。因此线性同余算法虽然速度较快，但安全性较低。

(3) 循环加密法。采用对计数器的输出进行加密,产生随机数作为会话密钥的方法。计数器的输出经主密钥加密后产生随机数,作为会话密钥,产生过程如图 11-4 所示。其中,C 为计数器的输出,加密算法模块用主密钥 km 对 $C+1$ 加密,加密后的结果为产生的随机数 X_i。

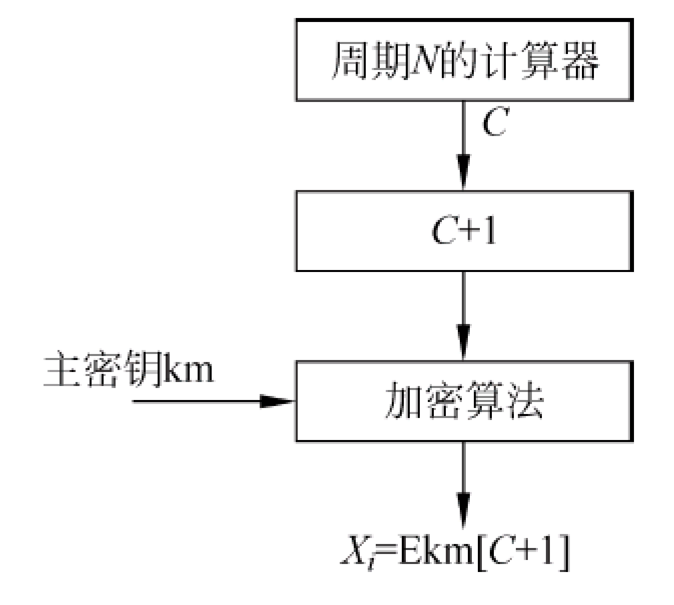


图 11-4 循环加密生成随机数

在密码学应用中,安全的随机序列需要满足三个基本特性:具有不可预测性;不能重复产生;能通过随机性统计测试。

随机性测试通常是通过检验被测随机序列是否满足随机序列的某些特性,来判定其是否随机的。测试方法通常根据随机序列所表现出的某些特性来设计,虽然通过某项随机性测试不能保证该序列是随机的,但没有通过某项测试,则说明了该随机序列不随机,由此可以发现随机数产生器设计或应用上的某些问题。

目前,已知的随机性检验方法有二百多种,而且,根据某些参数的改变,一种检验方法还有多种变形。其中比较有代表性的如美国商务部国家标准技术协会(National Institute of Standards and Technology, NIST)于 2001 年 5 月公布的 Federal Information Processing Standards 140-2(FIPS 140-2)标准和 Special Publication 800-22(SP 800-22)标准,德国资讯安全联合办公室(Federal Office for Security in Information Technology,BSI)于 2001 年 9 月发布的 AIS 31 标准。智能卡行业知名公司如英飞凌,意法半导体,恩智浦等的芯片均通过了 FIPS 140-2 和 SP 800-22 等标准对随机数发生器的测试。

FIPS 140-2:加密模块的安全需求(Security Requirements for Cryptographic Modules)中提出了建议的 4 种随机数测试方法,如表 11-5 所示。

表 11-5 FIPS 140-2 中规定的 4 种测试方法

序号	测 试 方 法	序号	测 试 方 法
1	Monobit 测试	3	游程测试
2	扑克牌测试	4	长游程测试

SP 800-22:密码应用中的真随机数和伪随机数统计测试套件(A Statistical Test Suite for Random and Pseudorandom Number Generator for Cryptographic Applications)中包括了测试随机数随机性的 16 种测试方法,每种测试针对随机序列中可能出现的非随机性进行了测试,评价方法统一采用 P-Value,将测试后得到的 P-Value 与显著性水平相比较,得到测试结果。表 11-6 列出的 SP 800-22 中提出的 16 种测试方法。

表 11-6 NIST SP 800-22 中规定的 16 种测试方法

序号	测 试 方 法	序号	测 试 方 法
1	频数测试	6	离散傅里叶变换测试
2	块内频数测试	7	非重叠模板匹配测试
3	游程测试	8	重叠模板匹配测试
4	块内最长游程测试	9	Maurer 通用统计测试
5	二元矩阵秩测试	10	Lempel-Ziv 压缩测试

续表

序号	测 试 方 法	序号	测 试 方 法
11	线性复杂度测试	14	累加和测试
12	序列测试	15	随机游动测试
13	近似熵测试	16	随机游动状态频数测试

AIS 31 测试标准：物理随机数产生器功能分类和评价标准(Functionality Classes and Evaluation Methodology for Physical Random Number Generators)及其提案中规定了 9 种随机数测试方法，如表 11-7 所示。

表 11-7　AIS 31 中规定的 9 种测试方法

序号	测 试 方 法	序号	测 试 方 法
1	解体测试	6	自相关测试
2	Monobit 测试	7	正态分布测试
3	扑克牌测试	8	多项式分布对比测试
4	游程测试	9	熵测试
5	长游程测试		

常见的随机数测试工具如依据 SP 800-22 标准的 NIST Statistical Test Suite(STS)软件，应用该测试工具可以发现随机数产生器设计和应用上的缺陷。依据智能卡随机数测试的需要，我们开发了一套随机数测试软件，软件综合了 FIPS 140-2、AIS 31 和 SP 800-22 中提出的方法进行测试，软件界面如图 11-5 所示。



图 11-5　随机数测试软件截图

11.3 传输协议测试

智能卡芯片与外部通信的接口分为接触式接口和非接触式接口，所以智能卡传输协议也分成接触式传输协议和非接触式传输协议两种类型(只是传输协议类型，具体传输协议还可分为多种，如符合 ISO/IEC 7816-3 的接触式卡，符合 ISO/IEC 7816-12 的 USB 接口的接触式卡)，对这两种类型分别进行测试。

11.3.1 测试平台的搭建

测试平台主要构件包括信号发生器、功率可调放大器、专用测试架、协议分析设备、数字示波器，这些设备均可和测试软件连接进行监测与控制。测试平台原理架构如图 11-6 所示。

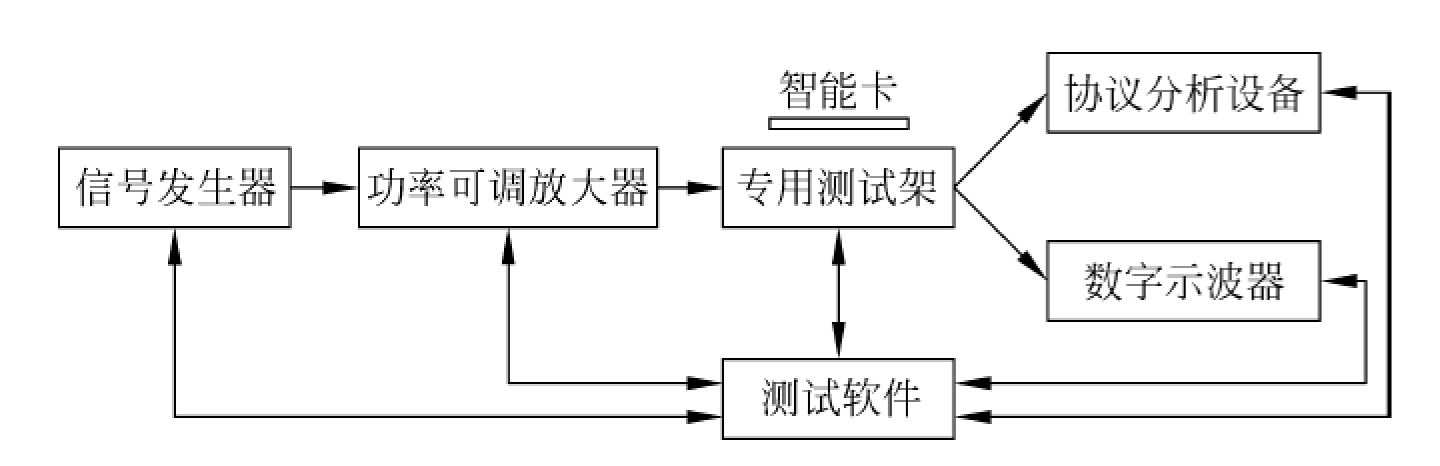


图 11-6 智能卡测试平台

在该平台中，可以测试如下内容：

(1) 射频能力和信号接口测试。

(2) 时序和帧格式测试。

(3) 通信协议测试。

(4) 卡片兼容性测试。

(5) 高层命令测试。

11.3.2 接触式协议测试

接触式智能卡的协议测试主要参考 ISO/IEC 7816-3 和 ISO/IEC 10373-3 测试标准，协议测试主要针对智能卡通信协议中的第三层和第四层。第三层中定义了传输协议的初始化，如复位、复位应答、协议和参数选择等内容。第四层中定义了智能卡的传输协议，常见的传输协议分为面向字符的 T=0 协议和面向分组的 T=1 协议。

ISO/IEC 10373-3 针对传输协议的第三层，对如复位应答时间、字符重发、奇偶校验差错检测、ATR 内容、字符等待时间(CWT)、分组保护时间(BGT)、ICC 对传输差错的反应、ICC 对协议差错的反应、重新同步等内容进行了测试。

在 ISO/IEC 7816-3 中对协议的第四层，T=0 和 T=1 两种传输方式进行了说明，并给出了 T=1 情况的部分测试用例。T=0 协议是面向字符的异步传输协议，每个字节都有相应的开始位，共八位数据，有奇偶校验位，命令总是由接口设备发送，在一个五字节头中告诉卡要执行什么命令。接口设备发送的命令头包括 CLA、INS、P1、P2、P3 五个连续的字节组成，其中 P3 为编码数据字节的数量，在命令执行期间传送这些数据字节。T=1 是面向分

组的异步传输协议，数据交换由字符部分和分组部分组成，字符部分在协议的第三层有详细的说明，分组包括三个字段：开始字段(prologue field)，信息字段(information field)和结尾字段(epilogue field)，而分组分为三个类型：信息分组(I-block)，接收准备分组(R-block)，管理分组(S-block)。与 T=0 协议不同，T=1 协议具有链接功能，允许接口设备或智能卡传送信息的长度大于 IFSD 或 IFSC 规定的长度。其中时序测试如图 11-7 所示。

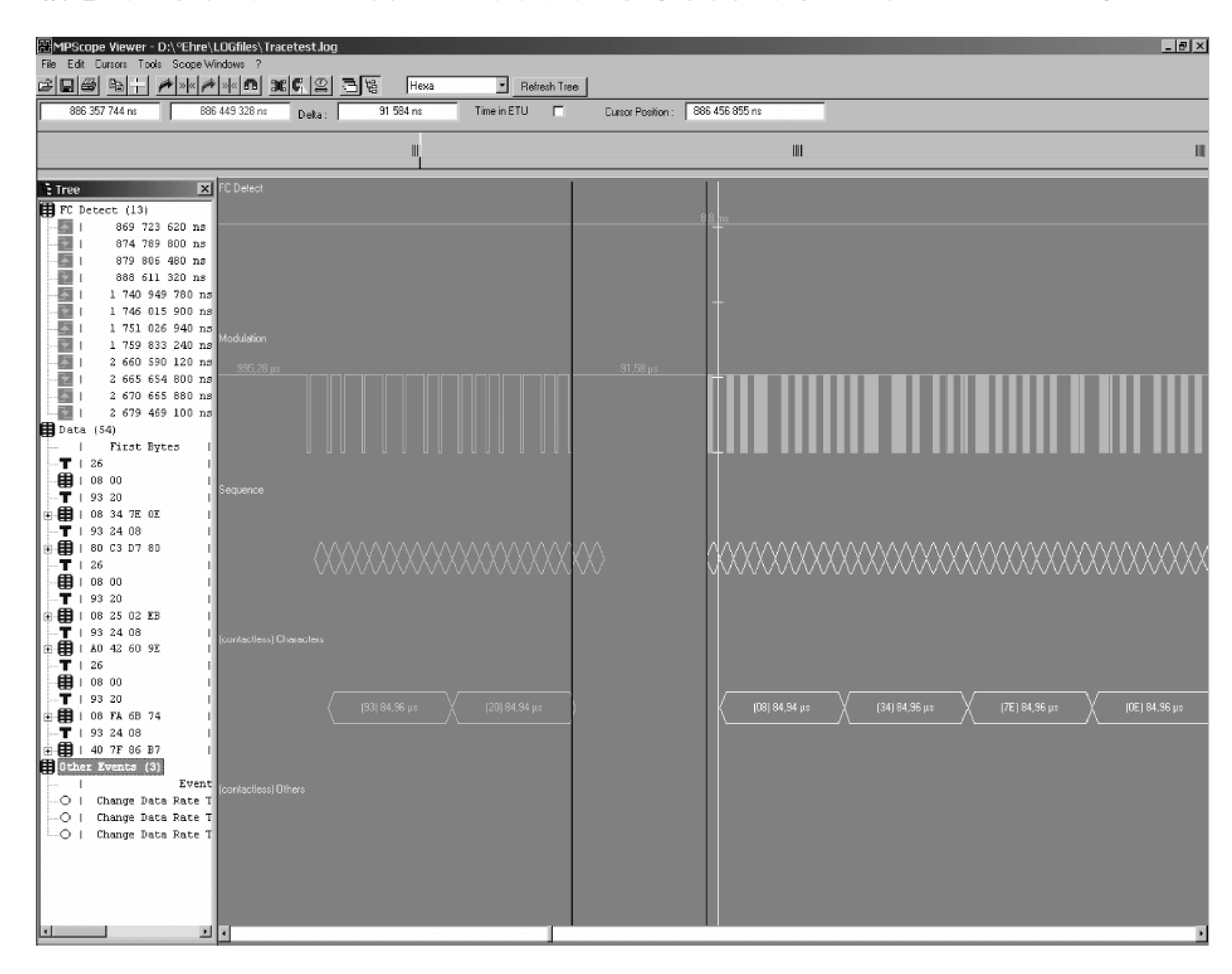


图 11-7　时序测试

11.3.3　非接触式协议测试

非接触式 IC 卡因作用距离的不同而分为三个不同类型的卡：

- 非接触式密耦合卡(CICC)，对应标准 ISO/IEC 10536-1～4；
- 非接触式近耦合卡(PICC)，对应标准 ISO/IEC 14443-1～4；
- 非接触式疏耦合卡(VICC)，对应标准 ISO/IEC 15693-1～3。

非接触式卡的协议测试包括对射频能量和信号接口的测试，以及对传输协议的测试两部分。其中最常用的 PICC 类型的卡片需要符合 ISO/IEC 14443 标准，在该标准中依据射频能量和信号接口的不同，传输协议可分为 Type A 和 Type B 两种，而每种卡的初始化和抗碰撞方法也不尽相同。Type A 和 Type B 的传输协议相同，是类似 T=1 的分组传输协议 T=CL。

对初始化和抗碰撞的测试主要参考 ISO/IEC 14443-3 和 ISO/IEC 10373-6 的增补 1，两个文档对 Type A 和 Type B 两种智能卡的初始化和抗碰撞与逻辑操作测试有详细的说明。

Type A 卡的初始化和抗碰撞测试包括对轮询(polling)的测试、状态机转换测试、抗碰

撞测试、RATS 处理测试、PPS 请求测试、FSD 处理测试等测试。对 Type B 卡的测试包括卡接收测试、状态机转换测试、抗碰撞测试、ATTRIB 处理测试和最大帧长度处理等测试。对 Type B 卡各种时序的测试参见图 11-8 所示。

Table G. 34—Type B specific timing table

No	Parameter	ISO Reference	Std min	Std Max	Measured value(s)
1	SOF low	ISO/IEC 14443-3：2001，7.1.4	10etu (～94.40μs)	11etu (～103.83μs)	
2	SOF high	ISO/IEC 14443-3：2001，7.1.4	2etu (～18.88μs)	3etu (～28.32μs)	
3	EOF low	ISO/IEC 14443-3：2001，7.1.5	10etu (～94.40μs)	11etu (～103.83μs)	
4	Bit boundaries	ISO/IEC 14443-3：2001，7.1.1	$\left(n-\frac{1}{8}\right)$etu	$\left(n+\frac{1}{8}\right)$etu	
5	EGT PICC to PCD	ISO/IEC 14443-3：2001，7.1.2	0μs	19μs	
6	TR0 for ATQB	ISO/IEC 14443-3：2001，7.1.6	$64/f_s$ (～75.52μs)	$256/f_s$ (～302.06μs)	
7	TR1 for ATQB	ISO/IEC 14443-3：2001，7.1.6	$80/f_s$ (～94.40μs)	$200/f_s$ (～235.99μs)	
8	TR0 for ATQB	ISO/IEC 14443-3：2001，7.1.6 ISO/IEC 14443-3：2001，7.10.3	$64/f_s$ (～75.52μs) or May be reduced	$(256/f_s)\times 2^{FWI}$ (～302.06μs×2^{FWI})	FWI= Max TR0=
9	TR1 Not ATQB	ISO/IEC 14443-3：2001，7，1.6 ISO/IEC 14443-3：2001，7，10，3	$80/f_s$ (～94.40μs) or may be reduced	$200/f_s$ (～235.99μs)	
10	Delay from the end of EOF and Subcarrier off	ISO/IEC 14443-3：2001，7.1.7	0etu	2etu	
11	Deactivation frame waiting time	ISO/IEC 14443-4：2001，8.1	$64/f_s+80/f_s$ (～169.92μs)	$65536/f_c$ (～4.8ms)	

Note：All timing values are calculated for carrier frequency f_c=13.56MHz and bit rate=f_c/128(～160kbit/s)

etu：elementary timing unit

图 11-8 ISO/IEC 10373-6 FPDAM1 中对时序的测试

逻辑操作测试包括卡对 ISO/IEC 14443-4 中的命令响应测试、卡错误检测处理测试、卡对 CID 的响应测试、卡对 NAD 的响应测试等。

11.4 COS 测试

智能卡 COS 的主要功能是实现对文件和安全机制的管理。通常，智能卡 COS 依照具体应用开发，同时也受到智能卡芯片性能的影响，因此 COS 是一个专用系统而非通用系统。

ISO/IEC 7816 和 ISO/IEC 14443 系列标准对智能卡的物理特性、传输协议、命令格式等进行了规范，但没有对 COS 的开发和设计形成统一的标准。目前，还没有任何一家公司的 COS 成为一种工业标准，这也为 COS 测试提出了很大的挑战。COS 从功能上可以划分为四个模块：通信管理模块、命令管理模块、安全管理模块和文件管理模块，如图 11-9 所示。

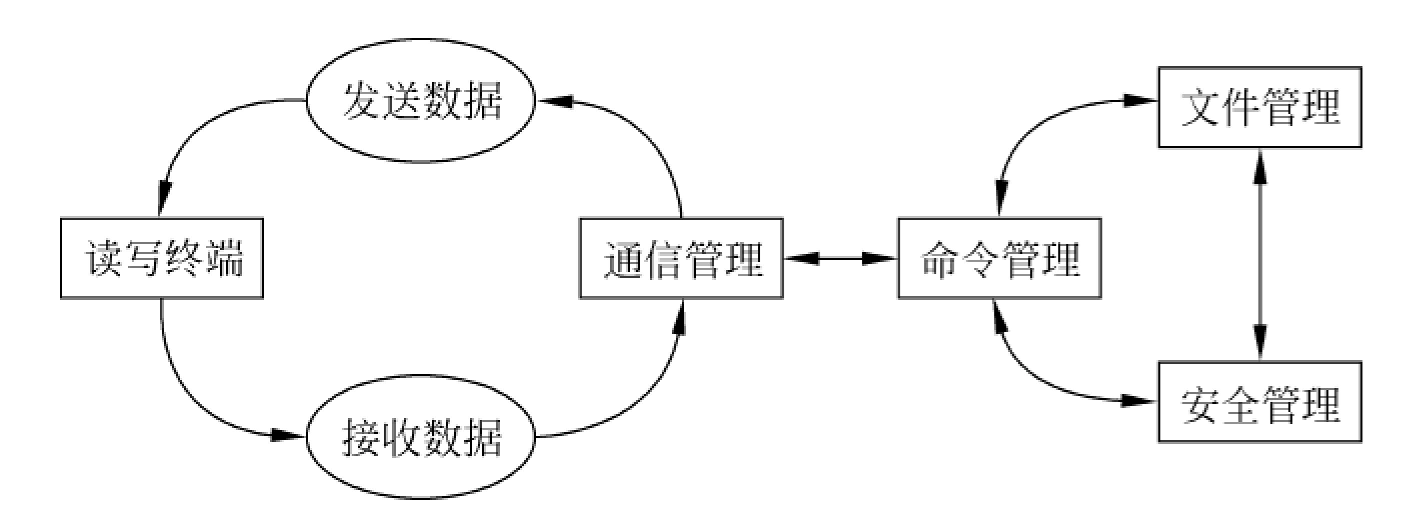


图 11-9　智能卡操作系统结构

COS 的基本功能测试主要参考 ISO/IEC 7816-4 标准，该标准中对智能卡的行业交换命令，安全和文件的逻辑结构等做出了详细的要求。对 COS 基本功能的测试均可以转换为对指令集的测试，通过向智能卡发送指令，对其返回值进行分析来实现。COS 从接收指令到返回响应的过程，依次经历了指令接收、指令解释、指令执行、返回数据和状态码的 4 个流程，如图 11-10 所示。智能卡在接收指令后，对指令进行解释，对指令的参数的合法性、应用的流程和当前的安全状态进行判断，当通过所有判断后执行指令，最后将执行指令后得到的数据和状态码返回。

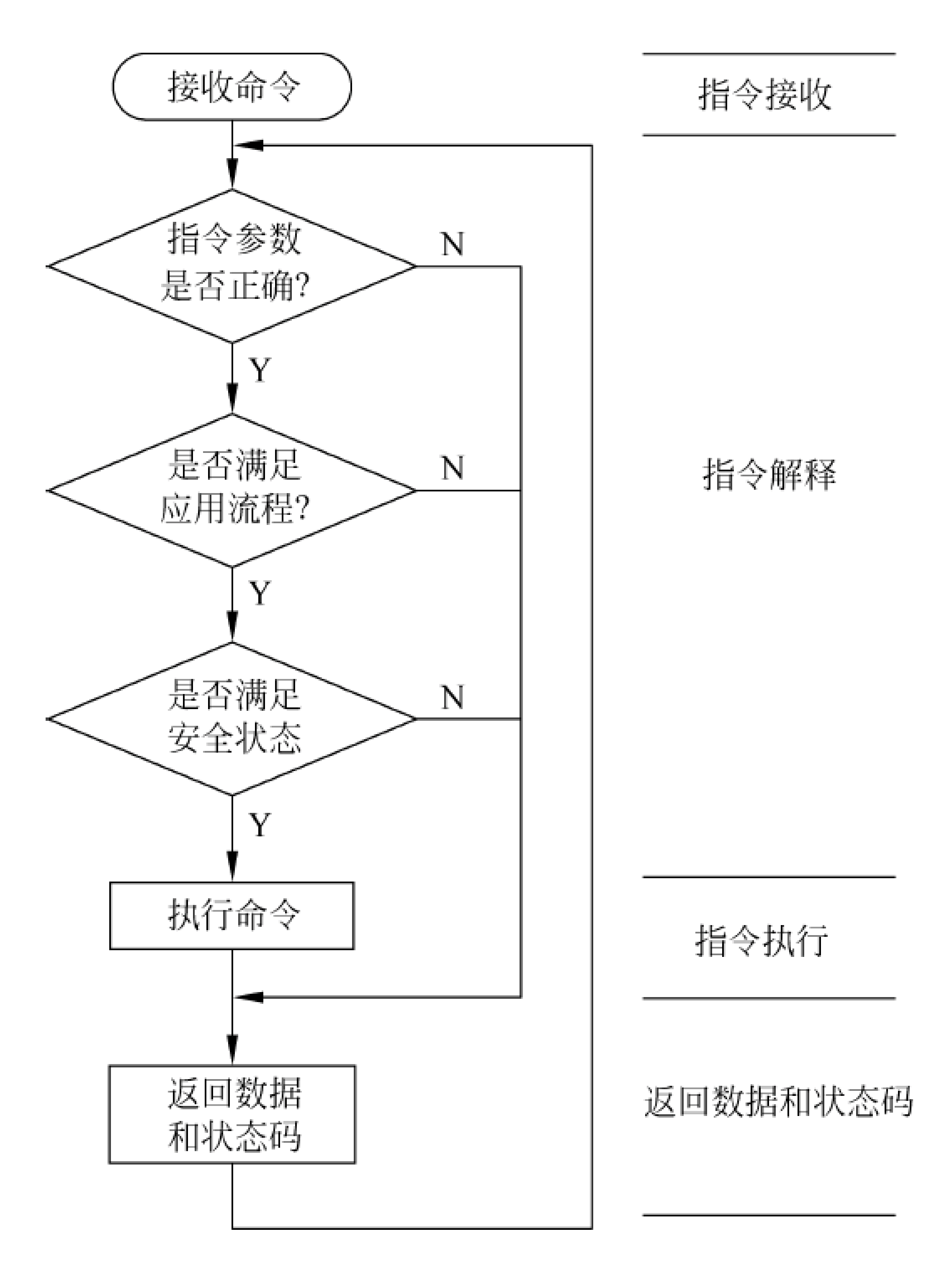


图 11-10　COS 处理指令的流程

从 COS 对指令的处理流程可知，COS 基本功能的测试可以分为 5 部分：指令参数测试、应用流程测试、安全状态测试、返回数据测试和状态码测试，而这 5 部分测试的关键在于指令是否能够实现预定的功能。通过指令参数测试，可以测试指令对于参数错误异常的处理能力，测试正常情况下指令是否能够正确地执行。应用流程和安全状态构成了指令执行时所处的环境，环境中很重要的组成部分是生命周期和安全状态，生命周期表明了卡片当前所处的应用状态，而安全状态决定当前应用的权限。指令总是同特定的应用和安全状态相关的，某条指令通常只能完成单一的功能，而在实际应用中，通常需要执行多条指令才能实现特定的功能，然而多条指令的执行又带来了应用流程的问题，不同的执行流程可能会导致截然不同的结果。每条指令的执行为后条指令的执行又创造了环境。返回数据和状态码是指令执行以后的结果，通过分析返回数据的格式和内容可以判断指令是否正确地执行，而状态码直观地给出了是否成功执行了指令，同时状态码也反映了指令的异常处理能力。

COS 基本功能测试的核心是指令功能的测试，而围绕指令的功能又可分为指令参数测试、应用流程测试、安全状态测试、返回数据测试和状态码测试 5 部分，本节将对这 5 种测试进行介绍。

11.4.1 指令测试

指令测试包括指令参数的测试、指令执行周期的测试。

在指令参数测试中，依据命令的数据组织格式，通过应用等价类划分方法将测试用例划分为若干等价类。如分为参数正确时的正向测试和参数错误时的反向测试。参数错误又可依据命令的结构分为命令错误、数据错误和 LE 错误，其中命令错误的错误类型包括缺项和参数错误两种，参数错误又可分为 CLA 错误，INS 错误等。对每种参数错误还可再细分，如 LC 错误可分为 LC 缺项、LC 过长、LC 过短、LC 边界值测试。上述测试用例的分类方法可以采用树形结构表示，如图 11-11，测试用例树的每个叶子节点说明了测试用例的设计方法。

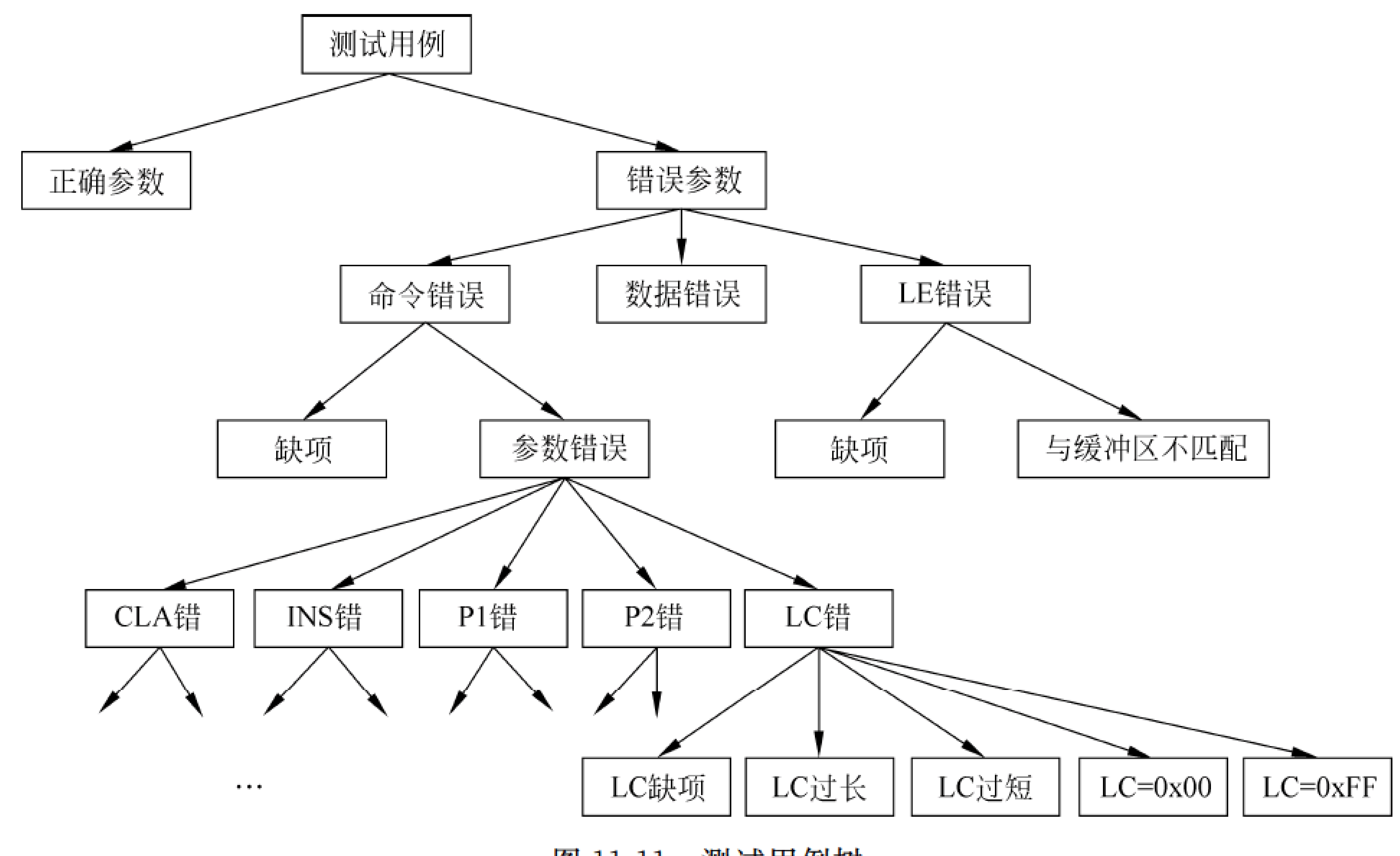


图 11-11 测试用例树

为了提高智能卡的安全性，通常将智能卡的工作状态划分为若干个生命周期，生命周期也就是智能卡所处的应用状态，每条指令只能在特定的生命周期下使用。对 COS 指令的测试与生命周期是相关的，因为对任何指令的测试都只能在特定的生命周期状态下进行。每条指令都有其应用的生命周期，只有在其有效的生命周期里才能够执行。例如错误认证超过一定次数以后，卡会锁定，进入失效期，在失效期的状态下某些指令不能执行，如创建和读取文件。卡片的生命周期分为创建期，安装期，使用期有效，使用期无效和终止期。创建期将传输密钥替换为系统密钥，并完成配置参数的写入；安装期完成智能卡文件系统和密钥文件的创建；使用期有效完成向智能卡文件中写入数据、配置和写入密钥等操作，智能卡发布给用户时，通常处在该状态；使用期无效：当用户由于非法操作，如认证次数超过限定次数等情况，导致智能卡进入使用期无效，在使用期无效，从安全性的角度考虑，不能进行如数据读取等操作；使用期终止：如智能卡由于应用到期等情况，需要终止应用，在使用期终止，智能卡将不能响应任何命令。

指令的生命周期测试是指令测试的一部分，每条指令都有其应用的生命周期(如第 5 章图 5-4 所示)范围。通过将卡配置到特定的生命周期，对指令的功能进行测试。观察在该生命周期下，指令的功能。如读取数据命令只能在使用期有效时使用，获取随机数命令可以在所有生命周期状态下使用。测试分为两部分：正向测试和反向测试。通过正向测试，检验指令在该生命周期下是否能够使用；反向测试，检验在该生命周期下对异常情况的处理。

11.4.2 应用流程测试

在实际应用中，智能卡实现某项功能，通常需要执行多条指令才能够实现，因此指令并不独立存在，各条指令之间互相依存，某条指令的执行可能需要满足另外一条命令执行所创造的前提条件。例如用于智能卡鉴别读写器真伪的外部认证流程分为如下步骤：首先由读卡器向智能卡发送一条产生随机数的命令，卡产生一个随机数并返回给读写器，读写器用预先存储的密钥对收到的随机数加密，将加密后得到的数据封装在外部认证命令之中并发送给卡；卡接收到外部认证命令后，从命令中取出认证数据，用预先存储的密钥解密，将解密后的明文与上一次产生的随机数相比较，若两者相同则通过认证，若不同则认证失败。外部认证的流程参见本书第 5 章。外部认证需要读写器首先发送获取随机数命令，然后再发送外部认证命令，两条命令的执行有一定的顺序，不能颠倒。针对外部认证流程可以设计多个测试用例，比如执行两次获取随机数命令，用第一次产生的随机数生成认证数据，进行外部认证，观察返回结果；又比如首先发送获取随机数命令，然后执行其他命令，如选择文件命令，最后用最先生成的随机数进行外部认证。

11.4.3 安全状态测试

智能卡指令的执行需要满足一定的安全状态，当满足指令要求的安全状态时，操作能够执行；而不满足指令的安全状态时，操作将被禁止。COS 通过安全状态控制指令的执行，实现对操作的安全管理。智能卡的安全状态与认证方式有关，通过进行相应的安全认证获得特定的安全状态。

常见的安全认证方式分为明文方式、外部认证和主动认证，具体描述如下：

(1) 明文方式，不需要通过任何的安全认证就可实现指令的功能。如果数据文件的访

问方式配置为明文方式,那么不需要任何安全认证,直接发送明文命令,就可实现对数据文件进行访问操作。明文访问方式的安全性较低,适用于对安全性要求不高,而对运行速度要求较高的场合。

(2) 外部认证方式,COS需要对读写机具提供的密钥进行认证,只有机具所持有的密钥与卡片中存储的密钥相一致的时候,才能通过外部认证。外部认证可以防止没有得到密钥的外部设备,如读写机具,对智能卡的操作。

(3) 主动认证方式,也称为内部认证,是外部设备对智能卡的一种认证机制,主动认证可以防止对智能卡的克隆。

安全状态与具体应用是相关的,例如读EF中存储的二进制数据应用,如果EF的访问方式为明文方式,那么无需任何安全认证,直接发送命令,就可获取EF中的数据;如果EF文件的访问方式配置为外部认证,那么必须用指定的密钥进行外部认证,通过认证后才能读取EF中的数据。安全状态测试也分为两类:正向测试和反向测试。正向测试对符合安全状态时的指令功能进行测试,观察指令是否能够实现预定的功能;反向测试则通过构造异常的情况,使得当前安全状态不满足实际需求,观察指令执行的结果。

11.4.4 返回数据和状态码测试

DATA是智能卡对命令APDU处理后返回给读写机具的数据,SW1和SW2为状态代码,直观地反映了程序执行的结果。返回数据与具体应用有关,应从数据格式和内容两方面进行测试。

对返回数据进行测试时,首先要对数据的格式进行测试,测试加密数据、状态码、校验值的组织顺序是否正确,测试TLV格式是否正确,比如TAG标签是否正确,Length标识的长度是否与真实的数据长度相符等。检查完数据的组织格式,需要对数据内容的正确性进行测试,如加密的返回数据经解密后,是否与应返回的明文相符合,CC是否正确等等。

状态码SW1和SW2是在智能卡接收到命令,对命令进行处理,返回执行的状态。状态码直观地反映了命令的处理结果。命令状态码由两个字节组成,状态字节的结构如图11-12所示。

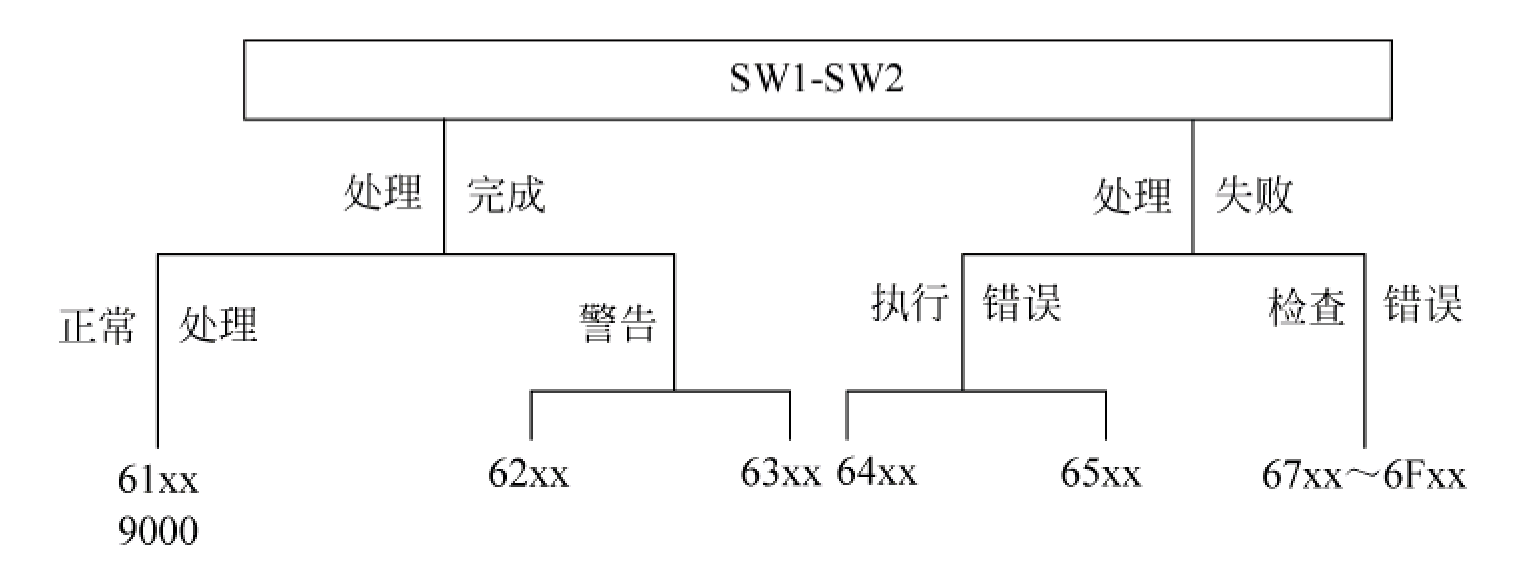


图11-12 状态码的组织结构

状态码分为处理完成和处理失败两类,而处理完成又分为正常处理和警告两种情况;处理失败分为执行错误和检查错误两种。进行状态码测试时,首先要依据ISO/IEC 7816-4中规定的返回命令值和COS开发商提供的技术手册,确定测试用例期望的返回状态码。如果测试用例返回的状态码与期望的结果相符则通过测试,若结果不符则测试失败。

11.5 应用业务测试

智能卡技术的应用领域十分广泛，如交通、金融、身份识别、电信、多媒体、医疗、数据存储等。而针对每个领域的应用都有相应的行业标准。如在身份识别领域，电子护照应用中，ICAO 在 DOC9303 中提出了对电子护照应用业务的需求，同时也提出了相应的测试标准。在金融领域，EMV 也提出针对借记卡和信用卡的行业标准及其测试标准。本节将以电子护照、SIM 卡和 EMV 卡的测试方法为例，介绍针对具体应用业务的测试方法。

11.5.1 电子护照测试

德国 BSI 机构开发了一套免费的电子护照测试工具 Golden Reader Tool，进行电子护照信息的读取和安全机制的判定，界面如图 11-13 所示。电子护照是否可用必须首先通过该软件的可读性测试。



图 11-13 Golden Reader Tool 2.9.2 软件截图

在完成可读性测试之后，ICAO 提出了针对机读旅行证件的一致性测试方法，由四部分组成，包括耐久性测试、安全通用标准(Security Common Criteria)、RF Protocol layer1-4 和 LDS Application layer6-7。

耐久性测试主要依据 ICAO 测试标准 Durability of Machine Readable Passports。该

标准对电子护照的物理特性进行测试,包括弯曲、温度、耐化学腐蚀、X 射线等测试内容。

安全通用标准(Security Common Criteria)在本章 11.7 节中有详细的说明。

RF Protocol layer1-4 和 LDS Application layer6-7 在 ICAO 发布的 RF Protocol and Application Test Standard for e-Passport 中有详细的说明,表 11-8 所示为该标准的组成,其中 RF Protocol layer1-4 对应第二部分能量和信号接口,协议的激活和传输协议测试;LDS Application layer6-7 对应第三部分,为应用协议和 LDS 测试。

表 11-8　RF Protocol and Application Test Standard for e-Passport

项　　目	说　　明	项　　目	说　　明
第一部分	架构和范围	第三部分	应用接口
第二部分	信号接口和 RF 协议	第四部分	PCD:信号接口和 RF 协议

第二部分信号接口和 RF 协议中将测试分为四个层次,包括:第一层,静电场测试,变化磁场测试;第二层,调制幅度测试,工作场强测试,通信稳定性测试,谐振频率测试;第三层的时序和帧测试包括登记测试,帧延迟时间测试,帧起始和帧终止测试,额外保护时间测试,SCIC SOF 之前时序测试,SCIC EOF 之后时序测试;第三层和第四层协议测试包括 Type A 和 Type B 的协议激活测试,数据交换协议测试等。

第三部分应用接口测试包括安全和命令测试,以及逻辑数据结构测试两部分。

测试用例采用如图 11-14 所示的形式表示。

3.3.1　Test Case 7816_C_1

Purpose	This function verifies the GetChallenge command(positive test)
References	ICAO PKI 1.1[R2]
Profile	BAC
Preconditions	The LDS application MUST be selected and basic access MUST be refused
Test scenario	1. Send the following GetChallenge APDU to the e-Passport. =>'00 84 00 00 08' 2. Send the same GetChallenge APDU to the e-Passport. =>'00 84 00 00 08'
Expected results	1. The e-Passport MUST return 8 random bytes and the status bytes '90 00' 2. The e-Passport MUST return 8 different random bytes and the status bytes '90 00'
Postconditions	Preconditions remain unchanged

图 11-14　互操作性测试用例

依照上述方法,公安部一所搭建了一套电子护照测试平台,并开发了一套电子护照自动测试软件。平台实景图如图 11-15 所示。



图 11-15　测试平台实景图

软件采用 PC/SC 体系结构设计,集成了测试程序运行、测试结果分析、测试报告生成等功能,为测试人员提供了统一的测试环境。可以自动完成命令测试、逻辑数据结构测试。软件运行时的截图如图 11-16 所示。其中标号 1 为测试控制区,可以实现执行测试程序、统计测试结果、读取 EF 数据、测试复位与清空记录等功能。标

号 2 为测试内容和配置信息设置区，可以选择测试用例、配置安全模式、选择机具等功能。标号 3 为测试运行记录区，记录了读卡器与电子护照间交互的命令和响应以及各用例的测试结果。标号 4 为测试结果统计区，记录了通过测试用例数、未通过用例数等信息。

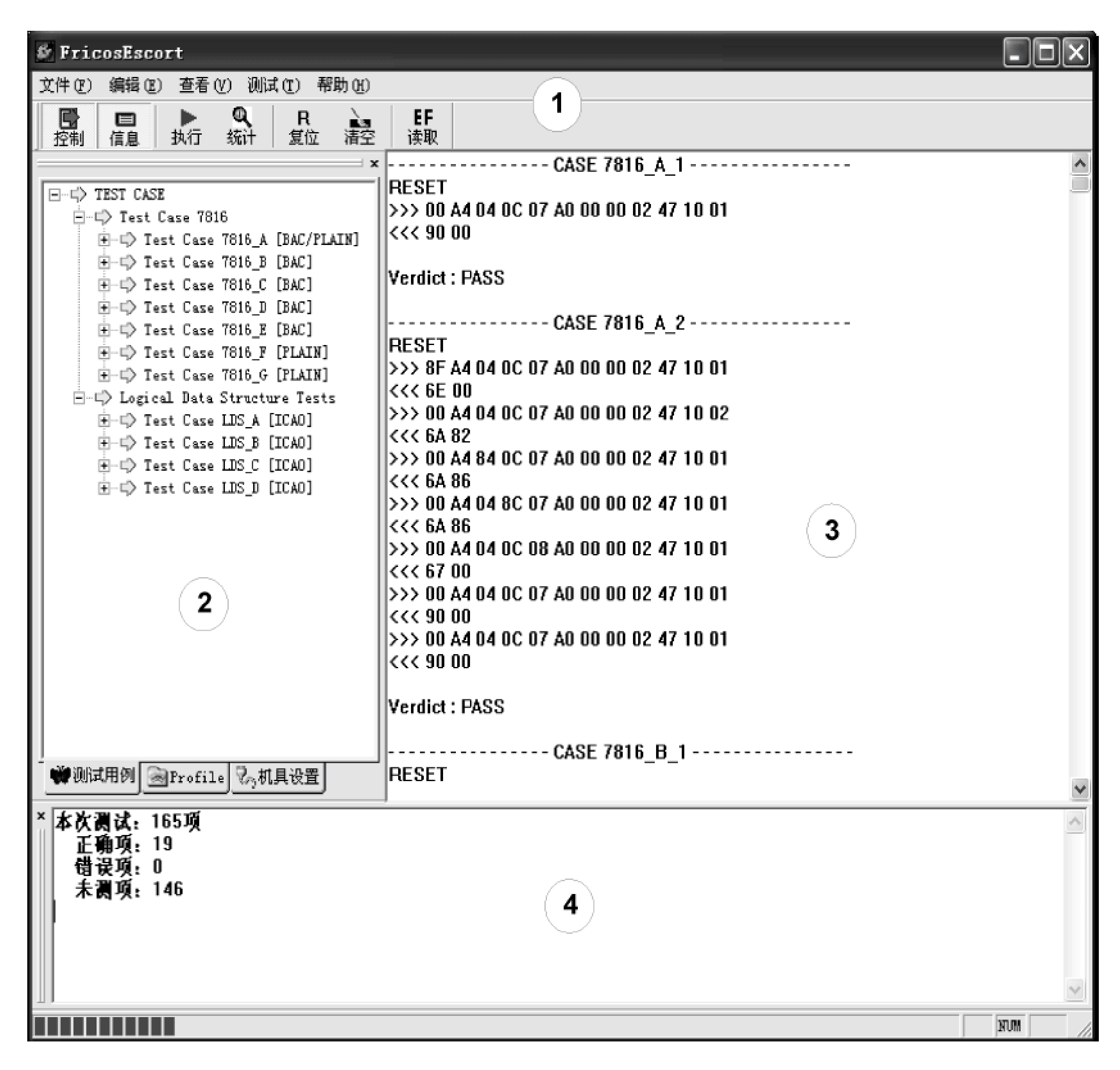


图 11-16 自动测试环境截图

COS 自动化测试软件可以自动完成从执行测试程序到生成测试报告的流程，无需人工干预，提高了测试效率和测试的准确性。

11.5.2 SIM 卡测试

SIM 卡是一种在无线通信领域广泛应用的智能卡。SIM 卡作为智能卡中特殊的一类卡，采用标准的接触式 IC 卡，同时它受到 ISO/IEC 7816 标准和 ETSI 的 GSM 11.11 等标准的规范，沿袭了智能卡在安全中的特色，对提高无线传输的安全性起到了重要的作用，在移动用户认证和移动商务中扮演重要的角色。

SIM 卡的测试可分为四个层次，包括物理层、传输协议层、COS 层和应用业务层。

物理层测试包括对 SIM 卡的物理及电气特性的测试，例如温度、湿度、存储器寿命、尺寸大小、电气特性等。

传输协议层测试包括对 SIM 卡传输协议的测试，如 ATR 返回数据的内容和结构测试，PTS(protocol type select，协议类型选择)测试，提高速度测试，异常处理测试等。

COS 层测试包括任务管理、文件系统、SIM 卡安全、APDU 解析等功能的测试。COS

掌管着 SIM 卡所有的软、硬件资源，所以对它的测试也最为关键。

应用业务层与具体应用相结合，包括对 STK(SIM tool kit，用户识别应用开发)能力、OTA(over-the-air，空中下载)能力、SCWS(smart card web server，智能卡网络服务)能力的测试等。

SIM 卡测试需要参考 GSM 11.11 规范，表 11-9 列出了该规范的主要内容，以供参考。

表 11-9 GSM 11.11 的主要内容

序 号	说 明	序 号	说 明
第一部分	物理特性	第五部分	功能描述
第二部分	电信号和传输协议	第六部分	命令描述
第三部分	逻辑模型	第七部分	基本文件的内容
第四部分	安全特性	第七部分	应用协议

11.5.3 EMV 测试

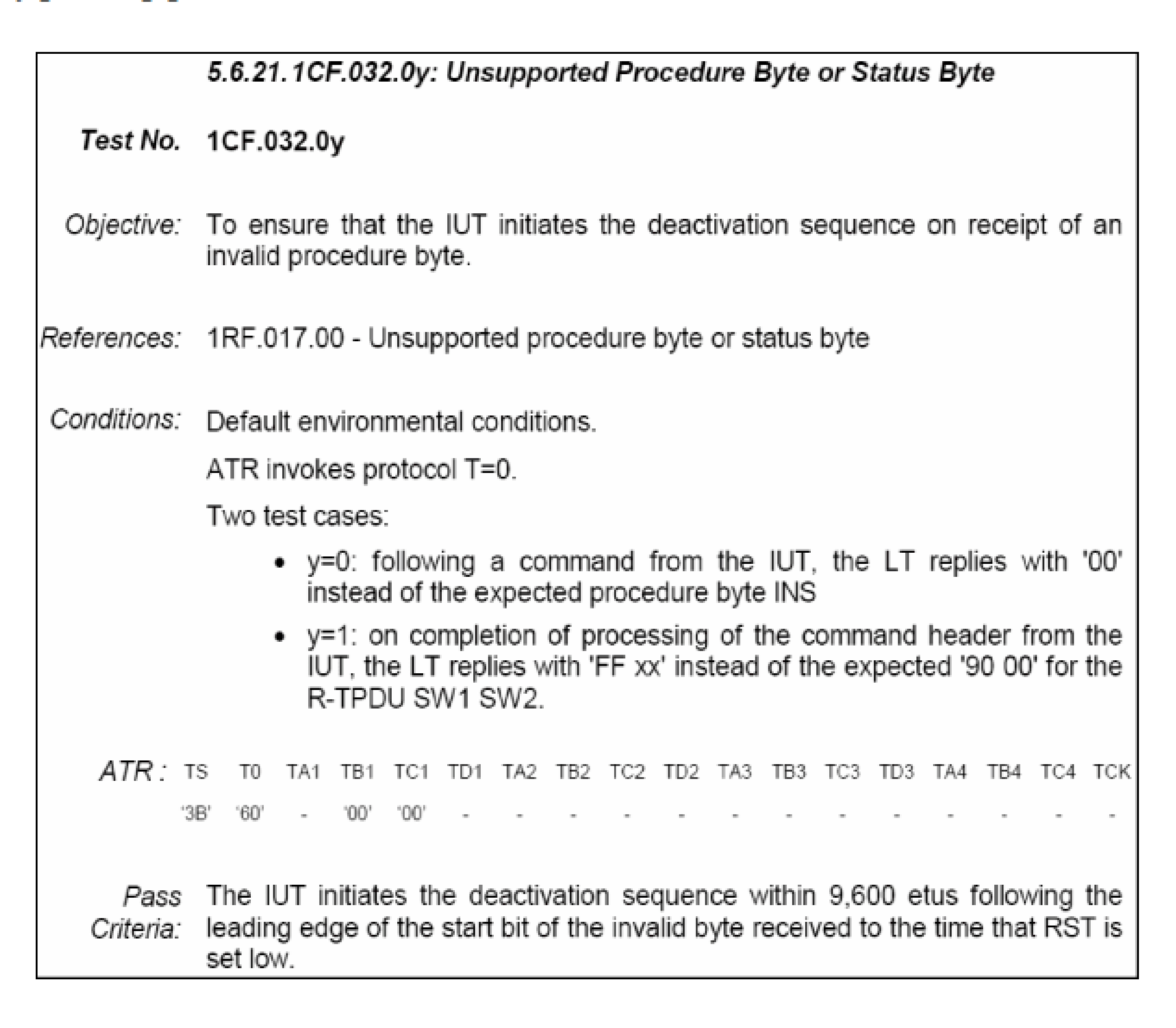
EMV 兼容性测试分为两个层次：

第一层覆盖了物理、电信号和传输协议的测试；

第二层覆盖了支付应用选择和信用卡交易流程的测试。

详细的测试方法请参考 EMVCo Type Approval Terminal Level 1 和 EMVCo Type Approval Terminal Level 2。EMVCo Level 1 测试分为智能卡的机械测试、电特性测试、复位应答测试、T=0 协议测试、T=1 协议测试、终端传输层测试等部分，每一部分都有对测试用例和方法的详细介绍。EMVCo Level 2 针对具体应用，分为支付系统的智能卡规范测试、应用选择测试、安全测试、数据对象测试、安全机制测试等。

以 EMVCo Type Approval Terminal Level 1 测试用例为例，如图 11-17 所示。



5.6.21. 1CF.032.0y: Unsupported Procedure Byte or Status Byte

Test No. **1CF.032.0y**

Objective: To ensure that the IUT initiates the deactivation sequence on receipt of an invalid procedure byte.

References: 1RF.017.00 - Unsupported procedure byte or status byte

Conditions: Default environmental conditions.

ATR invokes protocol T=0.

Two test cases:

- y=0: following a command from the IUT, the LT replies with '00' instead of the expected procedure byte INS
- y=1: on completion of processing of the command header from the IUT, the LT replies with 'FF xx' instead of the expected '90 00' for the R-TPDU SW1 SW2.

ATR:

TS	T0	TA1	TB1	TC1	TD1	TA2	TB2	TC2	TD2	TA3	TB3	TC3	TD3	TA4	TB4	TC4	TCK
'3B'	'60'	-	'00'	'00'	-	-	-	-	-	-	-	-	-	-	-	-	-

Pass Criteria: The IUT initiates the deactivation sequence within 9,600 etus following the leading edge of the start bit of the invalid byte received to the time that RST is set low.

图 11-17 EMV Level 1 测试用例

11.6 测试自动化

11.6.1 测试工具的开发

COS 的测试流程可分为五个步骤：设计测试用例，编写测试程序，执行测试程序，分析测试结果，生成测试报告，如图 11-18 所示。

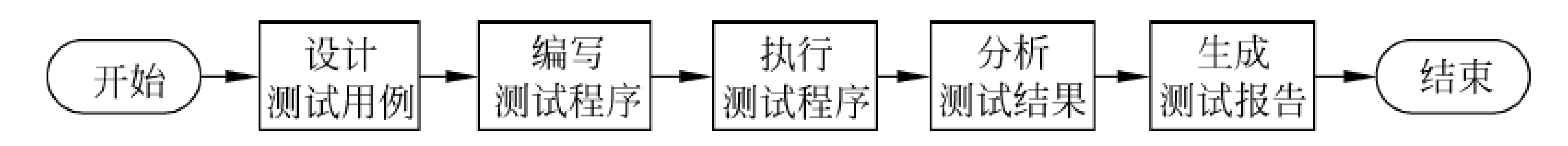


图 11-18 测试流程图

测试流程中第一个环节是测试用例的设计。测试用例的设计可以分成五部分，包括测试目的、测试前提条件、测试流程、期望结果、测试后状态。其中，测试目的说明了测试的主要内容、功能、目标等；前提条件说明执行该测试用例时需要满足的前提条件；测试流程详细说明了测试过程中各步骤执行时需要满足的条件和执行命令的流程；期望结果说明测试通过后，各测试步骤的期望结果；测试后状态说明了测试通过后应该满足的状态。测试用例可以采用模板来表示，如表 11-10 所示。

表 11-10 测试用例模板

项　　目	测 试 记 录	项　　目	测 试 记 录
测试目的	—	期望结果	—
前提条件	—	测试后状态	—
测试流程	—		

完成测试用例的设计以后，需要将测试用例转化为可以运行的测试程序，也可称为测试脚本，进而通过执行测试程序达到测试的目的。从测试用例到测试程序存在着一定的映射关系，并且测试程序与测试的执行环境密切相关，编写的测试程序必须在测试环境中能够正确并且高效的运行。目前国际上著名的智能卡测试软件有比利时的 Integri 软件、荷兰的 Collis 软件等，然而购买上述软件需要不菲的费用，为此公安部一所针对智能卡测试开发了一套智能卡测试环境和一套类 C 的测试用例描述语言。

测试用例描述语言支持大整数变量的定义，以及算术运算、逻辑运算、条件判断、循环结构等常见的程序结构，同时集成了如字符串比较、常见的加解密运算等功能的函数接口。表 11-11 所示为测试用例描述语言的说明。

表 11-11 测试用例描述语言

功　　能	说　　明
变量定义	采用 VAR 关键字定义变量
算术运算	支持加、减、乘、除四种算术运算和赋值运算
逻辑运算	大于、小于、等于三种逻辑运算
条件判断	支持采用 IF 关键字和逻辑运算结合的条件判断功能
循环结构	支持 IF 和 GOTO 结合的循环结构
常见功能	采用函数接口的形式，支持常见功能如比较、DES、RSA 加解密等功能

将测试用例转换为测试用例描述语言，需要在测试环境中执行，图 11-19 所示为公安部一所开发的 COS 测试软件。COS 测试软件集成了测试脚本编写和运行，智能卡读写机具的选择，记录的保存和显示等功能。通过将测试脚本输入到左侧的测试脚本编辑框，然后执行测试脚本，执行结果就可以在右侧的测试结果输出框中显示。智能卡测试的实践表明采用自行开发的智能卡 COS 测试用例描述语言和测试软件可以显著地提高测试工作效率，并且省去了购买国外软件的费用，具有经济实用、功能丰富多样的特点，是 COS 测试工作中不可或缺的利器。

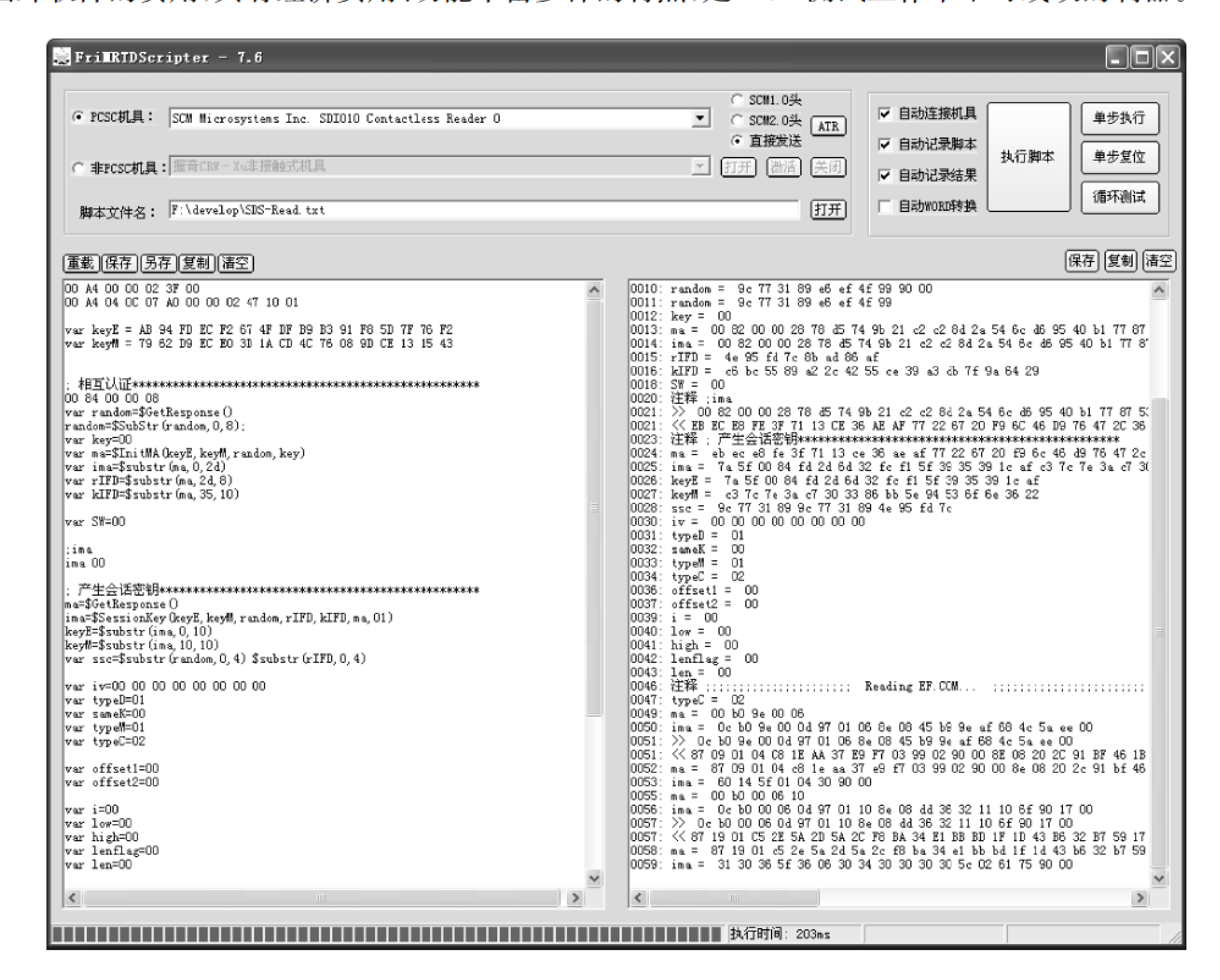


图 11-19　COS 测试软件

11.6.2　测试脚本的管理

测试脚本的管理在提高测试效率和实现脚本复用上有着重要的意义。通常一个 COS 开发项目会形成多个基线的 COS 版本。所谓基线指的是软件文档或源码的一个稳定版本。它是进一步开发的基础；是项目在某一时间的快照（snapshot），可以用来建立里程碑（milestone）。当某个 COS 产品形成稳定版本以后则构成了该 COS 的一个基线版本。针对基线版本的测试被称为基线测试，基线测试的所有测试用例构成了基线测试用例库。当需求发生变化或发现了 COS 基线版本中存在的问题时，则需要对 COS 进行修改。修改后，除了要考虑 COS 中新代码的正确性，还要考虑到新加入代码对原有代码的影响。因此修改 COS 后，需要对其所有的功能进行测试，还要补充新的测试用例。为了验证修改后 COS 正确性的测试被称为回归测试，而回归测试中所用到的测试用例构成了回归测试用例库。对测试用例库的维护是测试工作中重要的一部分，它包括 3 部分：删除过时和冗余的测试用

例，修改有缺陷的测试用例，添加新的测试用例。通过对测试用例库的维护来丰富和完善测试用例，实现从基线测试用例库到回归测试用例库的迁移。

测试用例的复用是提高测试效率、简化测试工作的重要手段。而测试用例的复用实质上需要测试用例具有较高的兼容性。要实现测试用例的兼容，需要形成统一的测试用例命名标准，脚本编写应该规范，应该有对测试用例的详细描述，如测试用例编写人员、编写日期、测试条件、测试流程、测试期望结果等。

11.6.3 COS 测试自动化

传统的手工测试方法，由于需要测试人员手工完成测试用例设计，编写测试程序，执行测试，测试结果分析等流程，耗费了大量时间，并且不可测的人为因素导致了测试准确性的降低。COS 的自动化测试成功地解决了上述问题，减少了测试工作量，加快了测试进度，提高了测试结果的准确性。

通常，测试用例的数量很多，需要运行大量的测试程序，对测试结果进行细致的分析，才能生成准确的测试报告。当测试发现问题，修改 COS 后，仍需对所有测试用例进行测试，以保证新版本 COS 的可靠性。重复性的测试工作，耗费了大量的时间，并且需要测试人员手工完成测试流程中的各个环节，人为的疏忽很容易导致测试准确性的降低。

半自动测试方法通过提供测试工具，辅助测试人员完成各环节的测试工作，但仍需要测试人员参与到测试流程的各个环节之中，测试工作量仍然相当可观。从对测试流程的分析可知，测试用例设计是测试中最重要的一个环节，而测试程序的编写则是测试用例设计思想的实现。增加测试用例数量是减少系统漏洞的重要方法，因此将测试人员从繁重的重复性工作中解脱出来，将精力集中在测试用例设计和测试程序编写上，是提高测试效率的行之有效的方法。自动测试方法正是从这个角度出发，通过为测试人员提供统一的测试环境，自动完成测试程序运行、测试结果分析和测试报告生成。测试人员仅需完成测试用例设计和程序编写两个阶段，其他环节则由计算机自动完成，无需人工干预，因此提高了测试效率，保证了测试的准确性。

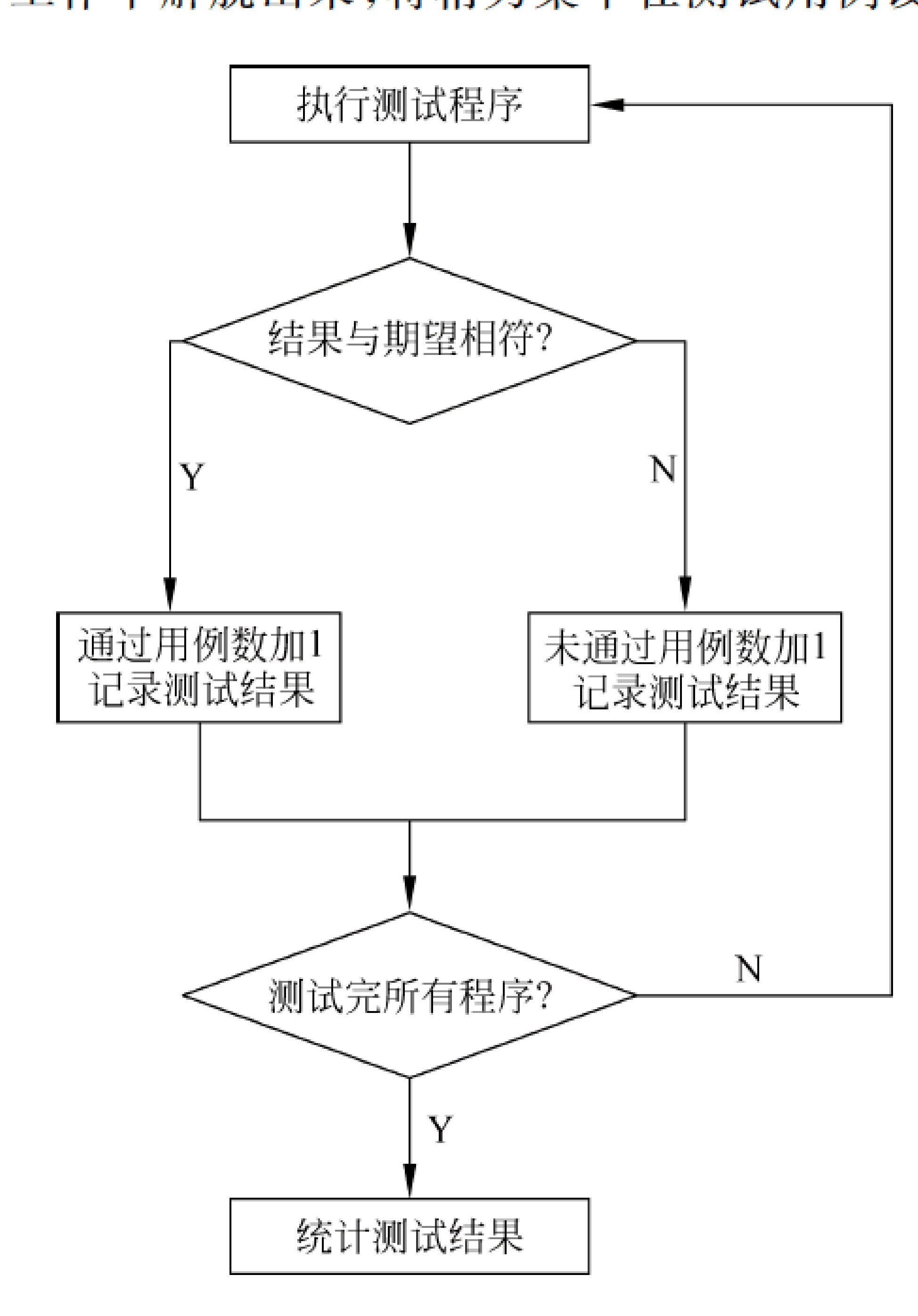


图 11-20 测试结果自动分析与统计

实际测试时，如果测试结果与期望的测试结果相符，则该测试用例通过，否则为未通过。测试环境通过将测试结果与预先设定的期望结果比较，来实现对测试结果的自动分析；通过对大量分析结果进行自动统计来生成测试报告。

测试环境中集成了脚本解释功能，将测试程序转换为实际的测试命令。对于反复测试的用例，也可将测试程序固化到测试环境中，便于执行。在执行测试程序后，需要对测试结果进行分析，如图 11-20 所示。测试集成环境自动将测试结果与期望结果进行比较，并记录运行记

录和测试结果,统计通过和未通过的测试用例数,当所有测试用例执行结束后,生成测试报告。

11.7 测评认证

11.7.1 介绍

测评认证包括测试、评估、认证三个不同又相互关联的概念。

测试是一种技术操作,它是按规定的程序和标准对给定的产品、材料、设备等的特性进行判定。评估是对测试/检验产生的测试数据、结果进行分析,形成结论的一种技术活动,测试和评估往往是合为一体的。认证是指第三方依据程序和标准对产品、过程或服务符合规定的要求给予书面保证(证书),用于评价产品质量和企业质量管理水平。认证以标准和测试、检验、评估的结果为依据。

对智能卡芯片、COS以及应用系统进行测试和评估,确定产品或者系统是否能够达到安全级别的可信程度。从事智能卡测评活动的第三方测评机构必须通过国家认证认可监督管理委员会对其测评能力的认可,符合ISO/IEC导则25-准和检验实验室能力的基本要求。

对智能卡产品进行测评认证具有重要的意义:有利于国家对智能卡产品行业和市场准入进行管理;推动智能卡行业相关技术和标准化的进程,促进智能卡产业健康发展不断进步与成熟;提高智能卡产品的市场竞争力,帮助生产厂商尽早发现问题,提升厂商的开发能力;使应用方了解智能卡产品满足的安全级别。

我国对信息安全产品的认证由中国信息安全认证中心负责完成。对智能卡芯片、COS以及应用系统进行认证是该中心代表国家对达到评价标准和标准要求的产品和系统进行的一种独立于用户、厂商以及测评机构之间的第三方的认可确认活动,即检验评估过程是否正确,并保证评估结果的正确性和权威性。中国信息安全认证中心满足国家认证认可监督管理委员会对其认证能力的认可需求,符合ISO/IEC导则65-实施产品认证制度的机构的基本要求。

11.7.2 强制认证

2008年中华人民共和国认证认可监督管理委员会在第3号公告中指出:

根据《中华人民共和国认证认可条例》、《强制性产品认证管理规定》、《强制性产品认证机构、检查机构和实验室管理办法》和《关于建立国家信息安全产品认证认可体系的通知》的规定,现做如下指定:

信息安全产品强制性认证指定认证机构:

(1)中国信息安全认证中心。

(2)信息安全产品强制性认证第一批指定实验室。

(3)信息产业部计算机安全技术检测中心。

(4)国家保密局涉密信息系统安全保密测评中心。

(5)公安部计算机信息系统安全产品质量监督检验中心。

(6)国家密码管理局商用密码检测中心。

(7) 中国信息安全测评中心信息安全实验室。

(8) 北京信息安全测评中心。

(9) 上海市信息安全测评认证中心。

2008 年中华人民共和国国家质量监督检验检疫总局和中华人民共和国国家认证认可监督管理委员会在第 7 号公告中指出：

根据《中华人民共和国产品质量法》、《中华人民共和国标准化法》、《中华人民共和国进出口商品检验法》、《中华人民共和国认证认可条例》、《强制性产品认证管理规定》和《关于建立国家信息安全产品认证认可体系的通知》，现决定对部分信息安全产品实施强制性认证。《第一批信息安全产品强制性认证目录》见表 11-12。

表 11-12 第一批信息安全产品强制性认证目录

产品类别	产品名称	产品的定义和适用范围
1. 边界安全	1) 防火墙	防火墙产品是指一个或一组在不同安全策略的网络或安全域之间实施网络访问控制的系统。 适用的产品范围为：(1)以防火墙功能为主体的软件或软硬件组合；(2)其他网络产品中的防火墙模块；不适用个人防火墙产品
	2) 网络安全隔离卡与线路选择器	网络安全隔离卡是指安装在计算机内部，能够使连接该计算机的多个独立的网络之间仍然保持物理隔离的设备。安全隔离线路选择器是与配套的安全隔离卡一起使用，适用于单网布线环境下，使同一计算机能够访问多个独立的网络，并且各网络仍然保持物理隔离的设备。 适用的产品范围为：(1)安全隔离计算机；(2)安全隔离卡；(3)安全隔离线路选择器
	3) 安全隔离与信息交换产品	安全隔离与信息交换产品是指能够保证不同网络之间在网络协议终止的基础上，通过安全通道在实现网络隔离的同时进行安全数据交换的软硬件组合。 适用的产品范围为：(1)安全隔离与信息交换产品；(2)安全隔离与文件单向传输产品
2. 通信安全	4) 安全路由器	安全路由器是指为保障所传输数据完整性、机密性、可用性，应用于重要信息系统的，具备 IKE 密钥协商能力，端口 IPSec 硬件线速加密能力的路由器。 适用的产品范围为分：集成了 IPSec(IP security)/SSL(secure socket layer)，以及防火墙、入侵检测、安全审计等一种或多种安全模块的路由器，仅接入公用电信网的路由器除外
3. 身份鉴别与访问控制	5) 智能卡 COS	智能卡芯片操作系统是指在智能卡芯片中存储和运行的、以保护存储在非易失性存储器中的应用数据或程序的机密性和完整性、控制智能卡芯片与外界信息交换为目的的嵌入式软件。 适用的产品范围为：(1)采用接触或/和非接触工作方式的智能卡的 COS；(2)其他被集成或内置了的 COS
4. 数据安全	6) 数据备份与恢复产品	数据备份与恢复产品是指实现和管理信息系统数据的备份和恢复过程的软件。 适用的产品范围为：独立的数据备份与恢复管理软件产品，不包括数据复制产品和持续数据保护产品

续表

产品类别	产品名称	产品的定义和适用范围
5. 基础平台	7) 安全操作系统	安全操作系统是指从系统设计、实现、使用和管理等各个阶段都遵循一套完整的系统安全策略,并实现了 GB 17859-1999《计算机信息系统等级保护划分准则》所确定的安全等级三级(含)以上的操作系统。 适用的产品范围为:(1)独立的安全操作系统软件产品;(2)集成或内置了安全操作系统的产品
	8) 安全数据库系统	安全数据库系统是指从系统设计、实现、使用和管理等各个阶段都遵循一套完整的系统安全策略,并实现 GB 17859-1999《计算机信息系统等级保护划分准则》所确定的安全等级三级(含)以上的数据库系统。 适用的产品范围为:(1)独立的安全数据库系统软件产品;(2)集成或内置了安全数据库系统的产品
6. 内容安全	9) 反垃圾邮件产品	反垃圾邮件产品是指对按照电子邮件标准协议实现的电子邮件系统中传递的垃圾邮件进行识别、过滤的软件或软硬件组合。 适用的产品范围为:(1)透明的反垃圾邮件网关;(2)基于转发的反垃圾邮件系统;(3)与邮件服务器一体的反垃圾邮件的邮件服务器;(4) 安装于已有邮件服务器上反垃圾邮件软件
7. 评估审计与监控	10) 入侵检测系统(IDS)	入侵检测系统指通过对计算机网络或计算机系统中的若干关键点收集信息并对其进行分析,发现违反安全策略的行为和被攻击迹象的软件或软硬件组合。 适用的产品范围为:(1)网络型入侵检测系统;(2)主机型入侵检测系统
	11) 网络脆弱性扫描产品	网络脆弱性扫描产品指利用扫描手段检测目标网络系统中可能被入侵者利用的脆弱性的软件或软硬件组合。 适用的产品范围为:网络型脆弱性扫描产品;不适用:主机型脆弱性扫描产品;数据库的脆弱性扫描产品;Web 应用的脆弱性扫描产品
	12) 安全审计产品	安全审计产品指能够对网络应用行为或信息系统的各种日志实行采集、分析,形成审计记录的软件或软硬件组合。 适用的产品范围为:将主机、服务器、网络、数据库及其他应用系统等一类或多类作为审计对象的产品
8. 应用安全	13) 网站恢复产品	网站恢复产品是对受保护的静态网页文件、动态脚本文件及目录的未授权更改及时地进行自动恢复的软件或软硬件组合。 适用的产品范围为:针对静态网页文件、动态脚本文件及目录进行自动恢复的产品

自 2009 年 5 月 1 日起,凡列入本强制性认证目录内的信息安全产品,未获得强制性产品认证证书和未加施中国强制性认证标志的,不得出厂、销售、进口或在其他经营活动中使用。

2009 年中华人民共和国国家质量监督检验检疫总局、中华人民共和国财政部、中华人民共和国国家认证认可监督管理委员会在第 33 号文件《关于调整信息安全产品强制性认证实施要求的公告》中指出:

国家质量监督检验检疫总局、国家认证认可监督管理委员会 2008 年第 7 号公告中涉及的信息安全产品强制性认证的强制实施时间延至 2010 年 5 月 1 日,在政府采购法规定的范围内强制实施。

11.7.3 分级评估

智能卡产品的分级评估是指依据国家标准 GB/T 18336-2008，综合考虑智能卡产品的预期应用环境，通过对整个智能卡的生命周期，包括设计、开发、管理、测试、交付等多个部分进行全面的安全性评估和测试，验证产品的保密性、完整性和可用性程度，确定产品对其预期应用而言是否具有足够的安全性，是否可以抵御使用中潜在的安全风险，以及是否满足相应评估保证级的要求。

智能卡行业内的安全评估标准与整个信息安全产品相同，从可信计算机系统评估准则(trusted computer system evaluation criteria, TCSEC)、信息技术安全评估准则(information technology system evaluation criteria，ITSEC)到信息技术安全评估通用准则(common criteria，CC)，CC 采用的标准为 ISO/IEC 15408，在我国等同为 GB/T 18336，即《信息安全 安全技术 信息技术安全性评估准则》。根据 CC，信息技术安全的评估由低到高划分为评估保证级 1(EAL1)—评估保证级 7(EAL7)七个级别。其中，每个高级别的 EAL 都比所有较低级别的 EAL 表达更多的保证，并通过保证组件的增强或增加来实现。

认证级别是对信息技术产品的安全性进行独立评估后所取得的安全保证等级，表明产品的安全性及可信度。获得的安全级别越高，安全性与可信度越高，产品可对抗更高级别的威胁，适用于较高的风险环境。此外，产品的认证级别并不表明通过认证的产品是绝对安全的。

根据申请的不同级别，需要递交的文档如表 11-13 所示。

表 11-13 各级所需文档

文档、保证级	EAL1	EAL2	EAL3	EAL4	EAL5
安全目的	√	√	√	√	√
功能规范	√	√	√	√	√
高层设计		√	√	√	√
低层设计				√	√
实现表示				√	√
对应性分析	√	√	√	√	√
安全策略模型				√	√
管理员指南	√	√	√	√	√
用户指南	√	√	√	√	√
功能测试		√	√	√	√
测试分析		√	√	√	√
生命周期模型			√	√	√
开发安全			√	√	√
工具与技术				√	√
交付与运行	√	√	√	√	√
配置管理	√	√	√	√	√
脆弱性分析		√	√	√	√

智能卡产品主要包括智能卡芯片和智能卡嵌入式软件，涉及移动电话的 SIM 卡、社会保障卡以及芯片操作系统(COS)等。目前国内厂商的很多产品都通过了 CC EAL4＋的安

全认证，例如由公安部第一研究所开发的中盾-eMRTDCOS，如图 11-21 所示。而国外已有很多的智能卡公司，例如 NXP、IFX、ST 的芯片都通过了 CC EAL5＋ 安全认证。



国家信息安全测评

信息技术产品安全测评证书

证书号：CNITSEC2009PRD0042

级别：EAL4+

北京中盾安全技术开发公司

（北京市海淀区首都体育馆南路 1 号　邮编：100048）

中盾－eMRTDCOS/V3.0

测评标准：GB/T 18336-2008《信息技术 安全技术 信息技术安全性评估准则》（idt ISO/IEC 15408:2005）

测评方法：《信息技术安全通用评估方法》（idt ISO/IEC 18045:2005）

测评报告编号：061CR20090429-EAL4+

此证书仅适用于该产品的上述版本或型号

签发日期：2009 年 10 月 21 日

有效期至：2012 年 10 月 20 日

批准人：

图 11-21　中盾-eMRTDCOS EAL4＋证书

11.7.4　测评步骤

智能卡产品的分级评估流程主要分为受理、预评估、评估和注册共 4 个阶段。

(1) 受理阶段

申请方向测评机构提出分级评估申请。由测评机构对申请方提交的申请书进行审查。

(2) 预评估阶段

受理完成后，测评机构对申请方提交的测评文档资料进行技术审查，来判定提交的资料内容是否符合要求。

(3) 评估阶段

申请方应按评估进度提交智能卡产品，一般在开始评估工作进行到 1/3 时。

测评机构的专业测评人员根据评估方案，严格遵照评估标准开展测评工作。

(4) 注册阶段

通过测评的产品，进行注册及颁发证书。

11.7.5　测评内容

测评主要包括文档评估、测试、现场核查共 3 个方面。

1. 文档评估

文件评估主要包括：

(1) ST(security target)评估。评估 ST 是否按照标准正确完备地定义了其安全功能,采用了哪些安全机制以及采取何种安全策略来有效对抗产品所面临的安全威胁、安全攻击以及应用环境。

(2) 配置管理文档的评估。确认是否合理使用了配置管理工具对产品的过程文档、代码、测试文档进行有效的管理,以及对 TOE(target of evaluation)的变更是否进行了记录和备份。

(3) 交付和运行文档的评估。评估交付和运行,即智能卡的整个生命周期内的活动是否进行了有效的控制。

(4) 开发文档的评估。对智能卡的对外接口命令、内部子系统、内部模块、源码等方面进行评估,确认是否合理有效,代码是否简洁、可读性强以及是否进行了详细的注释。

(5) 指导性文档的评估。对智能卡产品的操作指南是否详细、完善进行评估。

(6) 生命周期支持文档的评估。确定开发人员在产品的开发以及与相关上游、下游团队之间使用安全程序的能力。

(7) 测试活动文档评估。确定开发人员是否依据设计目标以及相应的标准对其产品进行了深入、认真、全面的测试,是否进行了详细的测试记录。

(8) 脆弱性评定文档的评估。确定已确认的产品脆弱性已经被描述并且是否进行了合理解释以及如何采用其他的管理、制度手段规避风险。

2. 安全性测试

安全性测试主要包括:

(1) 独立性测试。测评机构对申请方提供的产品按照申请方提供的测试用例进行验证,检验该产品是否能够正确实现所提供的安全功能。

(2) 穿透性测试。测评机构根据申请方提供的脆弱性分析文档,采用非常规的测试手段对产品进行测试。

3. 现场核查

现场核查主要包括:

(1) 核查配置管理。确认产品的开发文档是否齐全,是否合理有效地使用了文档管理软件进行管理,同时文档的管理是否符合国家的质量管理体系。

(2) 开发安全。确认产品的研发环境是否采取了防止产品机密信息泄露的安全措施,开发办公室是否进行了有效的监控管理,以及开发环境之外环境的安全。与各级研发人员进行交流,确认研发人员对所开发产品的设计思路是否清楚,对产品的相关技术、功能特点、安全措施是否明确,是否对文档进行了管理,是否满足质量管理体系的要求。

(3) 交付运行。确认从开发和内部测试到最终交付给最终用户所经历的移交过程都是安全的。

申请 EAL3 级(含)以上的产品需要进行现场核查。现场核查一般在评估过程进行至60%~80%进行。现场核查结束后,测评机构对核查的情况做总结,并提出不足以及需要更改的项目。申请方在规定的时间内按照现场核查的报告进行整改,然后向测评机构提交报告,整个测评工作结束。

11.7.6 测评标准

我国智能卡的安全测评机构需要经过国家认证认可监督管理委员会授权，安全测评根据以下的标准进行。

(1) GB/T 18336.1—2008。信息技术 安全技术 信息技术安全性评估准则 第 1 部分：简介和一般模型。

(2) GB/T 18336.2—2008。信息技术 安全技术 信息技术安全性评估准则 第 2 部分：安全功能要求。

(3) GB/T 18336.3—2001。信息技术 安全技术 信息技术安全性评估准则 第 3 部分：安全保证要求。

(4) GB/T 20276—2006。信息安全技术 智能卡嵌入式软件安全技术要求(EAL4 增强级)。

(5) GB/Z20283—2006。信息安全技术 保护轮廓和安全目标的产生指南。

参考文献

[1] 王育民，刘建伟. 通信网的安全：理论与技术. 西安：西安电子科技大学出版社，1999

[2] 王爱英. 智能卡技术. 第 2 版. 北京：清华大学出版社，2000

[3] 李鹤田. 信息安全认证与信息测评概念剖析. 信息安全与通信保密，2002，(5)：65～66

[4] 陈晓桦. 信息安全等级保护与分级认证. 计算机安全，2005，(02)：56～59

[5] 于扬. 智能卡操作系统测试技术研究与应用. 重庆：重庆大学硕士论文，2006

[6] 李国俊，王鸿娴. 信息安全评估与智能卡产品 EAL4＋测评. 信息技术与标准化，2008，(10)：23～26

[7] 李胜广，薛艺泽，张之津. 电子护照 COS 测试软件的设计与实现. 2008 年公安部第一研究所论文集，2008，26～30

[8] 张小波，李胜广，李莉等. 电子护照 COS 自动测试的研究与实现. 2009 年年公安部第一研究所论文集，2009，150～154

[9] 李莉，李胜广，张小波等. 电子护照应用测试平台设计与实现. 2009 年年公安部第一研究所论文集，2009，156～160

[10] International Standard ISO/IEC 7816-1. Identification Cards. Integrated Circuit Cards with Contacts，Part1：Physical Characteristics，1998

[11] International Standard ISO/IEC 7816-2. Identification Cards. Integrated Circuit Cards with Contacts，Part2：Dimensions and Location of the Contacts，1999

[12] International Standard ISO/IEC 7816-2. Identification Cards. Integrated Circuit Cards，Part3：Cards with Contacts-Electrical Interface and Transmission Protocols，2006

[13] Intern ational Standard ISO/IEC 7816-2. Identification Cards. Integrated Circuit Cards，Part4：Organization，Security and Commands for Interchange，2004

[14] International Standard ISO/IEC 14443-1. Identification Cards. Contactless Integrated Circuit Card-Proximity Cards. Part1：Physical Characteristics，1997

[15] International Standard ISO/IEC 14443-2. Identification Cards. Contactless Integrated Circuit Card-Proximity Cards，Part2：Radio Frequency Power and Signal Interface，2000

[16] International Standard ISO/IEC 14443-3. Identification Cards. Contactless Integrated Circuit Card-Proximity Cards，Part3：Initialization and Anti-collision，2000

[17] Internati onal Standard ISO/IEC 14443-4. Identification Cards. Contactless Integrated Circuit

Card Proximity Cards, Part4: Transmission Protocol, 2000
[18] International Standard ISO/IEC 10373-1. Identification Cards. Test Methods, Part1: General Characteristics, 2006
[19] International Standard ISO/IEC 10373-2. Identification Cards. Test Methods, Part2: Cards with Magnetic Stripes, 2006
[20] International Standard ISO/IEC 10373-5. Identification Cards. Test Methods, Part5: Optical Memory Cards, 2006
[21] International Standard ISO/IEC 10373-6. Identification Cards. Test Methods, Part6: Proximity Cards, 2001
[22] International Standard ISO/IEC 10373-6. Identification Cards. Test Methods, Part6: Proximity cards. Amendment 1: Protocol Test Methods for Proximity Cards, 2005
[23] International Standard ISO/IEC 10373-6. Identification Cards. Test Methods, Part6: Proximity Cards. Amendment 2: Improved RF Test Methods, 2002
[24] International Standard ISO/IEC 10373-6. Identification Cards. Test methods, Part6: Proximity Cards. Amendment 3: Protocol Test Methods for Proximity Coupling devices, 2004
[25] International Standard ISO/IEC 10373-6. Identification Cards. Test methods, Part6: Proximity Cards. Amendment 4: Additional Test Methods for PCD RF Interface and PICC Alternating Filed Exposure, 2005
[26] International Standard ISO/IEC 10373-6. Identification Cards. Test methods, Part6: Proximity Cards. Amendment 5: Bit Rates of fc/64, fc/32 and fc/16, 2005
[27] NIST Special Publication 800-22. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, 2008
[28] BSI Application Notes and Interpretation of the Scheme 31. Functionality Classes and Evaluation methodology for Physical Random Number Generators, 2001
[29] FIPS Publication 140-2. Security Requirements for Cryptographic Modules, 2002
[30] ICAO DOC 9309. Machine Readable Travel Documents, Part 1: Machine Readable Passport, Volume 1: Passport with machine Readable Data Stored in Optical Character Recognition Format, 2006
[31] ICAO DOC 9309. Machine Readable Travel Documents, Part 2: Machine Readable Passport, Volume 2: Specification for Electronically Enabled Passports with Biometric Identification Capability, 2006
[32] ICAO Machine Readable Travel Documents Technical Report. Durability of Machine Readable Passports, Version 3.2, 2006
[33] ICAO Machine Readable Travel Documents Technical Report. RF Protocol and Application Test Standard for e-Passport Part2: Tests for Air Interface, Initialisation, Anticollision and Transport Protocol, Version 1.02, 2007
[34] ICAO Machine Readable Travel Documents Technical Report. RF Protocol and Application Test Standard for e-Passport Part3: Tests for Application Protocol and Logical Data Structure, Version 1.01, 2007
[35] ICAO Machine Readable Travel Documents Technical Report. RF Protocol and Application Test Standard for e-Passport Part4: e-Passport Reader Tests for Air Interface, Initialisation, Anticollision and Transport Protocol, Version 1.01, 2007
[36] EMV Integrated Circuit Card Specifications for Payment Systems Book1: Application Independent ICC to Terminal Interface Requirements, Version 4.1, 2004
[37] EMV Integrated Circuit Card Specifications for Payment Systems Book 2: Security and Key Management, Version 4.1, 2004
[38] EMV Integrated Circuit Card Specifications for Payment Systems Book 3: Application Specification,

Version 4.1,2004

[39] EMV Integrated Circuit Card Specifications for Payment Systems Book 4: Cardholder, Attendant, and Acquirer Interface Requirements, Version 4.1,2004

[40] EMVCo Type Approval Terminal Level1 Test Cases, Version 2.1,2009

[41] EMVCo Type Approval Terminal Level2 Test Cases, Version 4.2b,2010

[42] GSM 11.11. Digital cellular telecommunications system(Phase 2+), Specification of the Subscriber Identity Module -Mobile Equipment(SIM-ME)interface,1995

[43] GSM 11.14. Digital cellular telecommunications system (Phase 2+), Specification of the SIM Application Toolkit for the Subscriber Identity Module - Mobile Equipment(SIM-ME)interface,1996



未来发展趋势

本章将从智能卡规范、智能卡发展方向以及物联网三个方面，探索智能卡领域未来发展的趋势。

智能卡技术的发展和卡规范的推行相辅相成，本章 12.1 节通过介绍四个应用非常广泛的智能卡规范，得出卡规范趋于一统的发展方向。

巴黎卡展是智能卡领域一年一度的盛会，卡展评选的芝麻大奖（Sesame Awards）是奖励参展智能卡公司的最先进、最富创意的设计，有些设计在一定程度上也会预示着未来智能卡的发展趋势。12.2 节对某些有创意的设计进行了介绍。

物联网是时下最火热的名词之一，它融合了当前最先进的技术，被喻为下一代网络技术的革命。物联网的发展也与未来人们的生活息息相关。物联网的基础是 RFID，与本书联系紧密。在 12.3 节，作者希望通过结合 RFID 技术介绍物联网来拓宽本书的知识层面，并开阔读者的视野。

12.1 卡规范趋于一统

12.1.1 半壁江山——GP 规范

GP 联盟是第一个制定跨不同行业的智能卡规范的组织，目前该联盟拥有包括金融机构、电信运营商、智能卡和终端制造商以及软件开发公司在内的 50 多家成员单位。该联盟由会员直接管理，并被划分为不同的委员会（即卡、终端、系统架构和商业开发等）。各个委员会负责 Global Platform 的开发和促进工作。

通过利用开放平台的倡议，GP 在智能卡领域发布了至关重要的智能卡标准——独立于硬件、厂商以及不受应用程序约束的卡管理规范。这个新规范提供了保护智能卡系统基础设施的安全和卡管理框架，它规定了包括智能卡、终端设备以及后端支持系统的一系列通信接口、安全框架、标准命令等实现方面的具体要求。

GP 规范是完全免费的，并已经在欧洲、北美、南美、亚洲和太平洋地区被许多政府机构和公司所采用。目前全球范围内有超过 25 项基于 GP 规范的智能应用和 7000 多万张 GP 卡片在流通。GP 规范在卡端的实现方面支持的智能卡技术包括 Sun 公司提出的 JavaCard 技术、微软提出的 Power Smart Card 技术和万事达公司提出的 Multos 技术。

GP 规范是一个灵活而强大的规范，为卡发行商创建多应用的芯片操作系统以满足商业需求的不断发展提供了极大的便利。GP 规范允许卡发行商利用当前的卡技术，如果未来出现新的技术，也不会对 GP 卡的结构造成重大影响。

1. GP 卡体系结构简介

GP 卡体系结构如图 12-1 所示，主要包括以下 7 个功能模块。

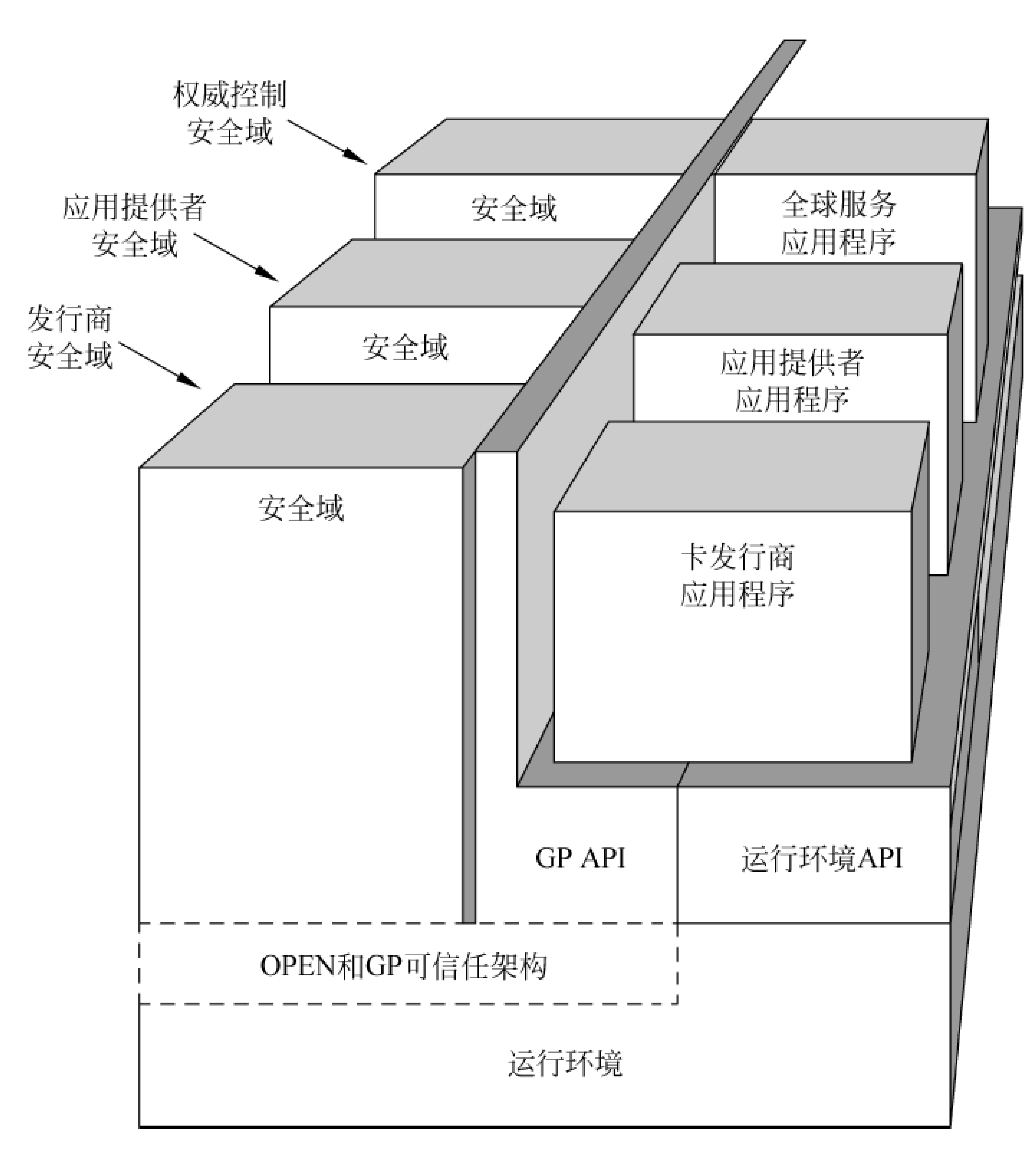


图 12-1 GP 卡体系结构

(1) 安全域(security domain)

安全域是特殊的应用程序。每一个应用程序和可执行加载文件都与一个安全域相关联。应用程序可以使用与之相关联的安全域所提供的加密服务，如个人化的支持、运行时安全报文的支持、发起或终止安全通道会话和数据块的加密和解密等。安全域主要有三类，每一类都是卡外实体在卡上的代表。

- 发行商安全域代表卡管理者，例如卡发行商，是卡片上第一个安装的应用程序，是所有安全域的基础，必不可少。
- 辅助安全域是附加可选的，它代表应用程序提供者、卡发行商或它们的代理商。
- 授权控制安全域是辅助安全域中特殊的一类，它扮演的是验证卡上的应用程序和代码是否符合安全政策的角色。

所有 GP 卡都必须有卡发行商安全域。拥有多个安全域的卡片允许应用程序提供者通过自己的安全域管理它的应用程序，并用它的密钥提供加密服务，而且这个密钥是完全与卡发行商分离的，不受其控制。

应用程序不需要知道它自己所关联的安全域，因为 GP 环境会提供应用程序与安全域的关联情况，而且由于引渡(extradition)，与应用程序相关联的安全域可能改变。引渡是指应用程序关联到其他不同的安全域的行为，例如可执行加载文件最初是和加载它的安全域

相关联,但随后它可能引渡到另一个安全域上。

(2) 全球服务应用程序(global service applications)

卡片上可以同时有多个全球服务应用程序,它可以为其他应用程序提供服务。全球服务应用程序与其他应用程序的区别在于拥有全球服务这个特权。当应用程序安装或注册时,它可以被分配包含一个或多个服务名称的全球服务参数。服务名称由两字节组成,第一个字节指示了其所属的服务系列,第二个字节指示了当前系列下服务的序列号。应用程序可以通过 GP 的 API 函数访问全球服务应用程序。

持卡人身份验证(cardholder verification method,CVM)程序就是一个典型的全球服务应用程序,当前支持的 CVM 是一个全局的个人识别码(PIN)。CVM 应用程序可以向卡上所有应用程序提供验证服务(包括找回 CVM 状态、找回剩余尝试次数、设定一个新的 CVM 值、验证 CVM 和设定最大尝试次数),也可以向授权的应用程序提供管理服务(包括 CVM 的注册、状态管理和 CVM 格式)。

(3) 运行环境(runtime environment,RTE)

卡片的 RTE 通过 API 函数为卡应用程序提供服务,这些服务可以帮助应用程序使用卡所具有的功能,例如确保卡上不同应用程序的代码和数据彼此安全地隔离、为卡和卡外实体提供通信服务等。

(4) 可信任架构(trusted framework)

应用程序接收的外部 APDU 命令首先要在安全域中去除封装,然后通过可信任架构的检验,投递到目标应用程序中。可信任架构需要检验:接收实体是否拥有可信任的授权路径、接收实体是否是安全域、目标应用程序是否存在、目标应用程序是否已经被其他逻辑通道选中(如果有多选限制)以及目标应用程序是否和当前选中的安全域相关联。

(5) GP 环境(global platform environment, OPEN)

OPEN 的主要职责是为应用程序的开发提供 GP 的 API 函数,例如 CVM、个人化和安全服务、锁卡和更新应用程序生命周期等。如果 RTE 没有提供这些功能,或者这些功能不符合 GP 规范,则由 OPEN 来实现这些功能。概括来讲,OPEN 主要有以下功能:

- 命令调度,包括应用程序和安全域的选择、逻辑通道管理(可选的)和卡内外通信的命令调度。
- 卡内容管理,包括卡内容的认证、安装、加载、删除和访问控制。
- 安全管理,包括安全域、应用程序与卡片的锁定和解锁、终止卡片、授权和日志的管理。
- 提供 GP 可信任架构。

(6) 卡内容管理(card context manage)

GP 规范所指的卡内容是卡中可执行载入的文件,这些文件可以存入:

- 不可改变的稳定内存(如 ROM),文件在卡片的制造生产阶段就被载入,不会被改变。
- 可变的稳定内存(如 EEPROM),文件可以在卡片的发行前或发行后装载或删除。

卡内容管理是指对卡内容进行加载、安装、引渡、注册更新和删除的操作。GP 为卡发行商及其合作伙伴提供了关于卡内容管理的最大的灵活性。GP 考虑到卡发行商不一定需要管理所有卡内容的变动,尤其是不属于卡发行商的内容,通过不同安全域的授权,卡内容

可以被批准改变。

卡内容管理器可以被看作以下三种实体：OPEN、发行商安全域以及持卡人认证方法服务提供者。

(7) 生命周期模型(life cycle model)

GP 规范还定义了生命周期模型，用来控制卡上各组件的安全性和功能性。具体内容包括卡片的生命周期模型、可执行文件/模块的生命周期模型和应用程序与安全域的生命周期模型。

卡片的生命周期可以分为以下几个状态：

准备操作状态(OP_READY)：此时 RTE 是有效的，卡发行商安全域被选中并随时准备接收、处理和响应 APDU 命令。

初始化状态(INITIALIZED)：本规范没有规定这个状态的功能。这个状态可能会将一些初始数据导入卡片，但是还没有做好将卡片发给用户的准备。从 OP_READY 状态转换到 INITIALIZED 状态的过程是不可逆的。

保护状态(SECURED)：处于这个状态的卡片向卡外实体表明发行商安全域的功能都已经准备完毕，可以接受发卡后的操作。从 INITIALIZED 状态转换到 SECURED 状态的过程是不可逆的。

卡锁定状态(CARD_LOCKED)：此状态禁止访问安全域和应用程序，也不能改变卡内容。从 SECURED 状态转换到 CARD_LOCKED 的过程是可逆的。只有当 OPEN、安全域或应用程序拥有锁卡权限的时候，才能使卡片从 SECURED 状态转换到 CARD_LOCKED 状态。

终止状态(TERMINATED)：此状态卡的绝大多数功能都被禁用，卡片只可能对 GET DATA 命令做出响应。从任何状态到 TERMINATED 状态的转换都是不可逆的。只有当 OPEN、安全域或应用程序拥有终止卡片权限的时候，才能使卡片转换到 CARD_LOCKED 状态。

可执行文件/模块只有一个状态——被加载状态(LOADED)。

而应用程序与安全域的生命周期可以分为以下几个状态：

安装状态(INSTALLED)：此状态下应用程序已经安装完毕且与相应的卡内组件链接完毕，而且此时应用程序没有被选中也没有进行初始化。

选择状态(SELECTABLE)：此状态下应用程序已经做好接收卡外实体的 APDU 命令的准备，此时应用程序的具体行为不在本规范中规定。从 INSTALLED 状态转换到 SELECTABLE 状态的过程是不可逆的。

锁定状态(LOCKED)：此状态下应用程序不能被选择。OPEN、应用程序本身、与应用程序相关联的安全域和其他拥有权限(global lock privilege)的应用程序或安全域可以将应用程序转换为锁定状态；与应用程序相关联的安全域和其他拥有权限的应用程序或安全域可以为应用程序解锁。

2. GP 安全体系结构简介

GP 的基本目标是确保卡片的 RTE、OPEN、发行商安全域、辅助安全域和应用程序在卡的整个生命周期内的安全性和完整性。GP 支持一系列的安全机制，如数据的完整性、资源的可利用性、机密性和认证。由于在智能卡应用领域中，卡片仅是其庞大的工程中的一小

部分，所以 GP 卡的安全性还依赖如代码测试、卡片的物理安全、安全密钥处理等方面，这些内容不在本规范中详述。

GP 详细规定了卡发行商、应用程序提供者和权威控制组织为卡安全所背负的责任，以及卡上各组件(运行环境、可信任架构、OPEN、安全域、全球服务应用程序、应用程序和终端系统)的安全需求。

GP 还规定了卡片对加密技术的支持，主要分为卡内容的安全管理和安全通信两方面。卡内容的安全管理是通过在文件中添加一部分冗余信息达到验证卡内容的完整性和真实性的目的，本规范介绍了以下 4 种卡内容安全管理的方法。

(1) 可加载文件数据块的散列运算(load file data block hash)。为了验证 GP 卡上的可加载文件的完整性。散列运算还被用在计算签名和授权管理标记中。

(2) 可加载文件数据块的签名(load file data block signature)。卡外实体对可加载文件数据块的散列值生成的一串鉴定值。该鉴定值由相应的安全域来验证。

(3) 授权管理标记(delegated management tokens)。当卡内容发生改变时，授权管理功能(由卡发行商提供)会对这一事件进行签名，生成授权管理标记。该标记由相应安全域来验证。

(4) 收据(receipts)。安全域对授权管理进行验证时会产生收据，用来证明卡内容已经被改变。

GP 所定义的安全通信是在 ISO 标准之上的一个广义定义，包括将 APDU 命令封装成安全报文、认证和安全信道的发起、运作和终止。安全通信可以为命令 APDU 提供以下服务：

(1) 通信实体的认证。通过加密数据的交换，卡或卡外实体可以证明对方的真实性。

(2) 信息的完整性与认证。接收实体可以确保发送数据的实体是经认证的，且数据没有被改变。

(3) 信息的机密性。确保传输中的数据不会被未经认证的实体得到。

GP 规范在江湖中的名声犹如它的名字 Global Platform 一样响亮，由此也可以看出它想要一统江湖的“野心”。在智能卡操作系统快速发展的今天，确实需要有人站出来规范操作系统，增强操作系统的通用性，GP 规范试图充当这一角色。在高端智能卡市场，许多卡片均遵循 GP 规范，如果说现在 GP 规范一统高端卡市场还为时尚早，那么至少 GP 规范已经坐拥高端卡市场的半壁江山。

12.1.2 百变金刚——JavaCard 规范

JavaCard 技术是 Java 语言编程和智能卡开发相结合的技术，它克服了智能卡软硬件开发技术过于专业、开发周期长的缺点，是在智能卡硬件系统的基础之上，在卡内通过软件构造的一个支持 Java 程序下载、安装和运行的软硬件系统。通过引入 Java 虚拟机技术，JavaCard 在保留了原有智能卡应用的便捷、安全等特性的同时，继承了 Java 技术的硬件无关特性。JavaCard 开发人员可以从供应商那里获得现成的 JavaCard 产品以及集成 Java 开发环境，使其能集中主要精力在应用的设计上，简化了开发复杂度，提高了应用的可重用性。

与传统的智能卡相比，JavaCard 技术拥有平台无关性、高灵活性、高安全性并支持一卡多应用等优点。纵观近些年全球智能卡市场，JavaCard 数量呈现急剧增长的势头，截止到

2007 年 5 月，在全球市场累积达到 30 亿张，这一市场份额远高于同样是开放式智能卡平台的 Multos 智能卡。

1. JavaCard 系统体系结构

JavaCard 系统体系结构如图 12-2 所示，主要包括以下功能模块。

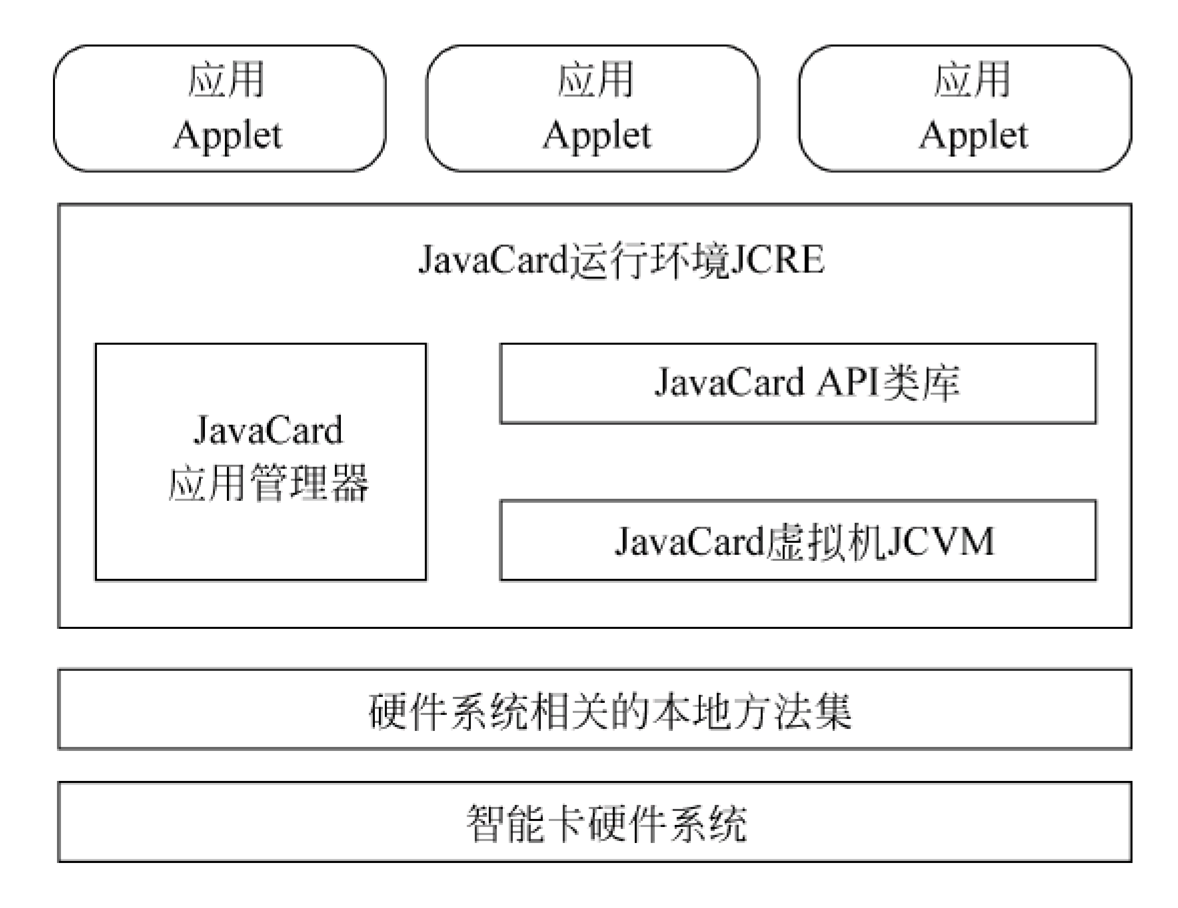


图 12-2　JavaCard 体系结构

(1) 智能卡硬件系统（hardware system）

包括 CPU、易失性存储器和非易失性存储、通信电路、加密协处理器等模块。

(2) 硬件系统相关的本地方法集(native methods)

完成基本的 I/O 通信、存储、加密等对硬件进行控制操作的本地方法。主要是为硬件提供了本地支持，如 EEPROM 空间的申请和访问，各种加解密算法提供的 C 语言接口等。

(3) JavaCard 虚拟机(JavaCard virtual machine，JCVM)

在智能卡硬件系统上通过软件构造的用于解释执行字节码的虚拟计算机。

(4) JavaCard API 类库(JavaCard application programming interface classes)

API 类库主要为开发人员定义了一整套编程接口类，描述了 API 包和类的核心和扩展，提供了编写智能卡应用程序所需要的接口。

(5) 应用管理器

包括安装管理器和删除管理器，分别负责卡上应用程序的安装和删除。这部分是可选的组件。

(6) JavaCard 运行环境(JavaCard runtime environment，JCRE)

包括 JavaCard 内虚拟机、本地方法、API 类库、应用管理器、事务处理和数据通信等。主要负责应用程序包的下载和安装，以及实现整个 JavaCard 的调度工作，其中包括初始化、通信和逻辑通道的管理等。

(7) JavaCard 应用程序(applet)

在 JavaCard 上运行的应用程序，简称 Applet。

2. JavaCard 技术规范简介

目前成熟的 JavaCard 技术规范的版本是 V2.2.2，最新的 V3.0 Classic 版本也已经发

布。JavaCard 技术规范由三部分组成：分别是 JavaCard 运行环境规范、JavaCard 虚拟机规范和 JavaCard 应用编程接口(API)规范。下面对这些规范进行简要介绍：

(1) JavaCard 运行环境规范

本规范的结构如图 12-3 所示，主要定义了 JavaCard 的运行期行为。

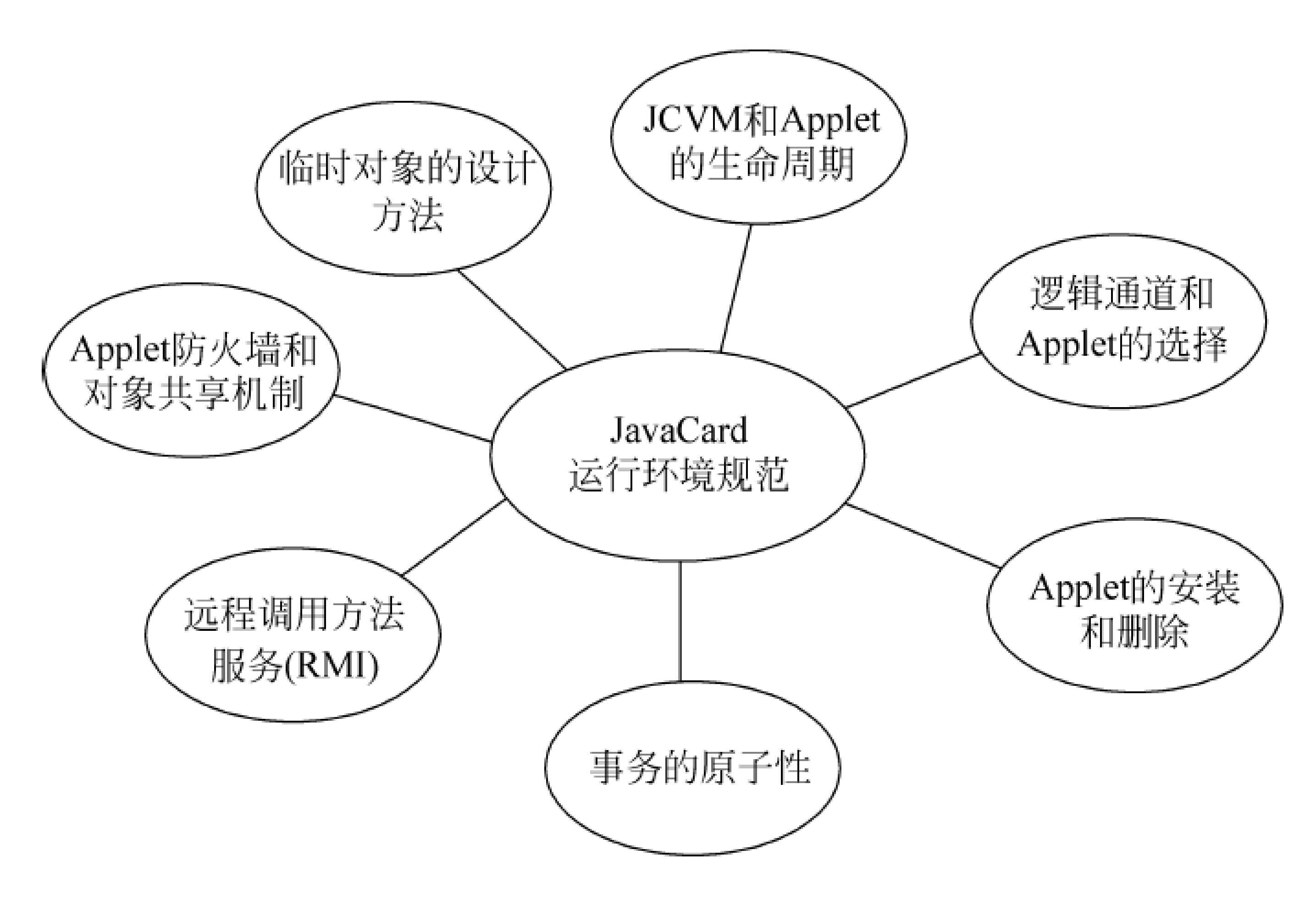


图 12-3 JavaCard 运行环境规范结构图

JCVM 的生命周期与卡片本身的生命周期一致，从卡片被制造测试并发行给持卡人前的某一时刻开始，直到卡片被废弃或损坏之时结束。当卡片掉电时，JCVM 并不停止运行，因为它的状态在卡的非易失内存中被保留下来。启动 JCVM 需要初始化 JCRE 并创建所有在 JCVM 整个生命周期中都处于激发状态的 JCRE 框架对象。在 JCVM 启动之后，所有与卡片的交互在原则上受卡片中某个 Applet 控制。当电源从卡片上移走时，所有包含在 RAM 中的数据全部丢失，但是存储在非易失内存中的状态仍然被保留下来。当电源重新供应时，JCVM 再一次被激发，此时 JCVM 的状态被恢复，并等待新的输入。

JavaCard 的 Applet 生命周期从它在运行环境中成功注册开始，直到它被 Applet 管理器删除时生命周期结束。运行环境可以通过调用 Applet 的公共方法(install()、select()、deselect()和 process())与 Applet 进行交互，实现相应功能，例如：安装 Applet 时，运行环境不会直接调用其构造函数，而是通过调用 install()方法，由 install()方法调用构造函数完成 Applet 实例的创建。

JavaCard 规范 V2.2.2 支持逻辑通道的概念。终端可以为卡片打开最多 20 个会话，每个会话过程对应一个逻辑通道。当一个 Applet 在一个逻辑通道上被激活后，它只接收在此逻辑通道上的后续的 APDU 命令，直到 Applet 被取消选择为止。编号为 0 的逻辑通道是基本逻辑通道，它在卡片复位时就被激活，其余的逻辑通道可以通过通道管理命令开启和关闭。

Applet 有时需要瞬态对象存储中间变量，JavaCard 平台不支持 Java 语言中的关键字 transient，但是可以通过以下方法实现存储中间变量：利用堆栈、局部变量、类的静态成员或者其他已存在对象的成员。

因为 JavaCard 允许卡上有多个 Applet，所以出于安全性的考虑，运行环境需要支持一

种机制：将不用的应用程序隔离，使其不能非法访问其他应用程序的数据，如果数据需要共享，应用程序需要提供相应的接口。在 JavaCard 中，应用的隔离是通过应用防火墙实现的。每个 Applet 被分配给一个执行上下文，用于控制对象的访问权限。一组执行上下文（即相同访问权限）包含多个 Applet 对象实例，它们之间相互访问是允许的，但是与另一组执行上下文之间相互访问会被防火墙禁止。当需要跨越防火墙访问其他应用的对象时，JavaCard 也提供了一些方便、安全的访问机制，如 JCRE 入口点对象、全局数组、JCRE 的访问权限和共享接口，这些访问机制不受防火墙的限制。

在特殊情况下，如果卡片在交易过程中失去电源，或者交易流程的错误操作都可能导致数据更新出现异常。所以 JCRE 需要保证事务（逻辑上对一组数据的操作）的原子性，即保证事务相关的所有数据的更新操作全部完成，或者全部不进行。JavaCard 平台提供了事务处理模型来保证其原子性。Applet 首先调用 JCSystem. beginTransaction 方法启动一个事物，此时并没有进行更新操作，直到应用程序调用 JCSystem. commitTransaction，所有数据更新操作提交完成。如果在调用 JCSystem. commitTransaction 前卡片掉电或其他系统错误发生，所有待更新数据恢复到之前的值；如果 Applet 内部发生错误或者用户需要取消事务，需要调用 JCSystem. abortTransaction 来撤销更新操作。

JavaCard 的远程方法调用（remote method invocation，RMI）是 Java 平台 RMI 技术的子集。服务器应用程序（JavaCard 上的 Applet）创建并访问远程对象，客户端应用程序（卡外实体的应用）获得对远程对象的引用，然后为这些对象调用远程方法。RMI 支持存储于不同地址空间的对象之间的彼此通信，该项技术已在第三代移动通信技术中的 Java SIM 卡中广泛使用。卡片的传输层为基于 RMI 的 Applet 提供程序包（RMIService 类中的 javacard. framework. service）作为服务请求。服务器和客户端应用程序的 APDU 交换过程被抽象化，不是直接通过 APDU 命令完成，而是通过处理对象的方式完成的。

Applet 的安装和删除是一个复杂的过程，在 JavaCard Platform V2. 2. 2 中定义了 Applet 的安装管理器和删除管理器的概念，指定了它们的最小需求。安装/删除管理器是一个可选的组件，它可以以 Applet 的形式出现，拥有与其他 Applet 一样的特点，也可以以其他形式实现，本规范并没有做强制性规定。

（2）JavaCard 虚拟机规范

本规范定义了 Applet 的下载方法、用于智能卡的 Java 编程语言和 Java 可兼容的虚拟机、二进制数据表示方法和文件格式以及 JCVM 指令集。本规范的结构如图 12. 4 所示。

开发人员编写完 Applet 源代码后，通过 Java 编译器生成 Java 类文件（也称字节码）。在准备将应用程序下载到智能卡之前，类文件通过转换器转换为 EXP 文件和 CAP 文件。EXP 文件即输出文件（export file），包含了 JavaCard 程序包的公共信息和连接信息；CAP 文件比 Java 类文件更加紧凑、高效。终端设备（例如 PC）的安装工具加载 CAP 文件，并将其复制到智能卡中。智能卡中的安装程序接收到 CAP 文件后开始进行 Applet 的安装与注册，之后虚拟机就可以运行该 Applet。

因为智能卡的资源有限，不可能实现 Java 平台的全部功能，所以 JavaCard 平台只拥有 Java 平台的一部分特性，并保留了 Java 语言面向对象的编程风格，非常适合在智能卡上进行程序开发。如图 12-4 所示，本规范定义了 JavaCard 对 Java 平台技术的支持情况和扩充，例如对 Java 语言特点的支持情况（包括对数据类型、对面向对象的特性、关键字以及 API 类

和接口的支持)、JCVM 的属性约束(对程序包、类、对象和方法等的约束)、JCVM 与 Java 虚拟机的异同和 RMI 的限制等。

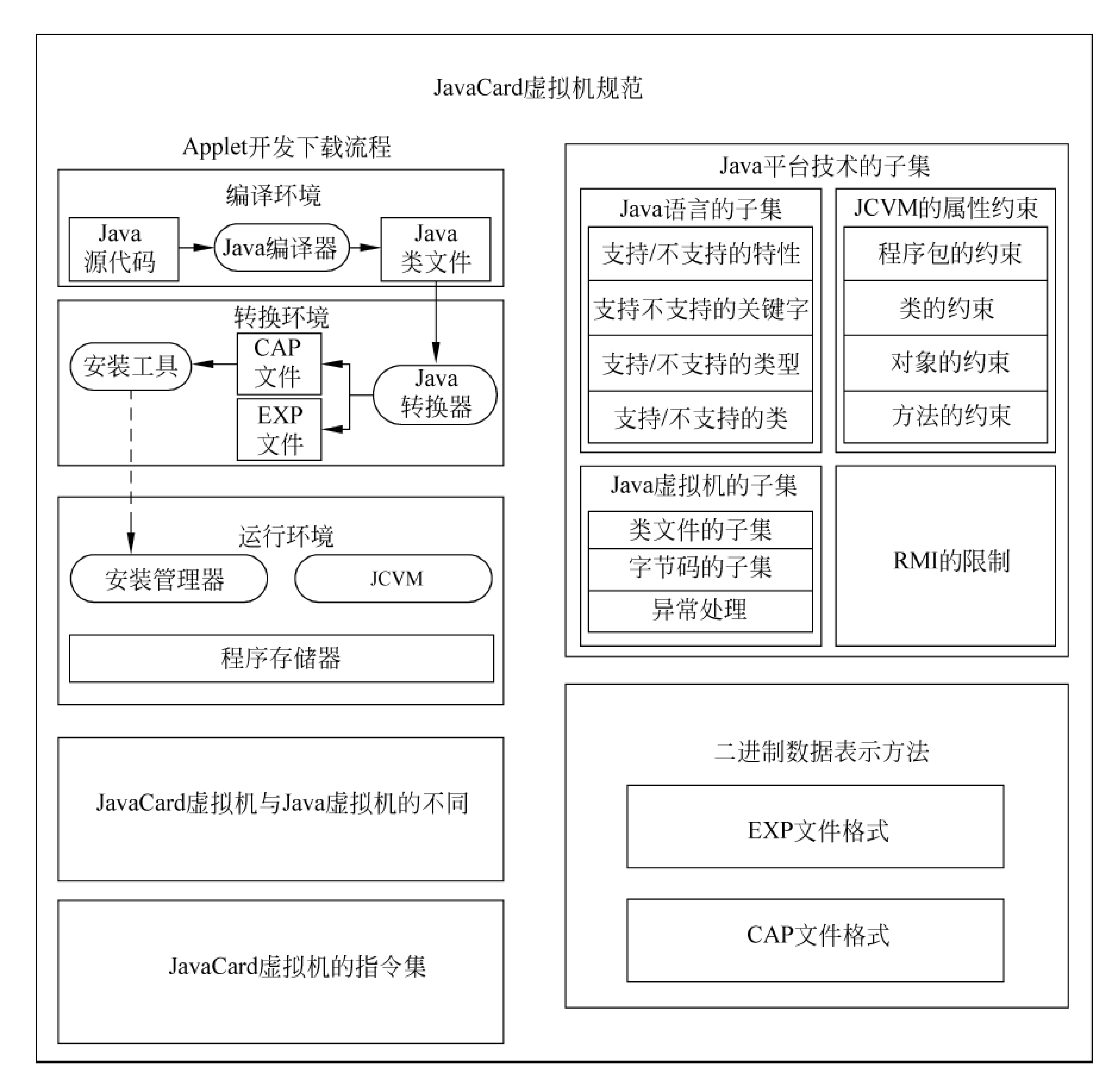


图 12-4 JavaCard 虚拟机规范结构图

(3) JavaCard 应用编程接口(API)规范

本规范定义了用于智能卡应用程序核心框架及扩展 Java 程序包和类,它是 Java 平台 API 的一个小型子集,还包括了自己专门支持 JavaCard 的 Applet 的核心类子集。

下面对这些程序包做一个简要介绍,方便读者快速了解各程序包内容和作用,更多的信息请读者查阅 JavaCard 应用编程接口(API)规范。

java.io 程序包:是标准 Java 编程语言包中 java.io 的一个子集。该程序包只包含一个异常类 IOException,在 I/O 发生异常时抛出异常信号。

java.lang 程序包:是由标准 Java 编程语言包中 java.lang 衍生而来,包含两个类 Object 和 Throwable,还有许多异常类。Object 是所有 JavaCard 平台的基类,Throwable 是所有错误和异常的基类。

javacard.rmi 程序包:定义了一个远程调用接口,它的方法可以被卡外实体的客户端程序调用。还定义了异常类 RemoteException,它是 IOException 的子类,在 RMI 发生异常时抛出异常信号。

javacard.framework 程序包：定义了 JavaCard 框架的接口、类和异常，为 JavaCard 环境提供了最小需求功能。该程序包中重要的类和接口如下：

- AID：这个类封装了与应用程序相关的应用程序标识符(AID)。
- APDU：提供控制卡输入输出的方法。
- ISO/IEC 7816：封装了许多 ISO/IEC 7816-3 和 ISO/IEC 7816-4 的参数。
- JCSystem：提供了许多控制系统功能的方法，如传输管理、瞬态对象、对象删除机制、资源管理器和卡内 Applet 对象共享机制。
- MultiSelectable：为支持逻辑通道功能的 Applet 提供方法。
- Shareable：接口共享。
- Util：提供各种用于操作数组和数组中元素的方法。

javacard.framework.service 程序包：提供了用于框架服务的接口和类。Dispatcher 类用于添加、移除服务的注册和 APDU 命令的调度；Service 接口包含了处理 APDU 命令的方法；RemoteService 提供了处理 RMI 的服务；SecurityService 提供了查询当前安全状态的服务；BasicService 为所有的服务提供了基本的实现方法。

javacard.security 程序包：提供了包含 JavaCard 上安全和加密相关的公用函数的类和接口，这些函数的功能包括：各种密钥的产生、数据摘要、随机数产生、签名和会话密钥交换等。

javacardx.apdu 程序包：该扩展包能够支持 ISO/IEC 7816 中定义的可选的 APDU 相关机制，包含了一个接口类 ExtendedLength，可以支持 APDU 的扩展长度机制。

javacardx.biometry 程序包：该扩展包提供了包含 JavaCard 上生物特征相关功能函数的类和接口。可以依靠这个程序包创建生物特征相关的服务器应用，并可以向卡上其他客户端应用提供服务。

javacardx.crypto 程序包：提供安全和加密的扩展程序包，包含一个接口和一个抽象类。接口类 KeyEncryption 提供了密钥更新的方法，Cipher 是加解密算法的抽象基类。

javacardx.external 程序包：该扩展包提供了访问存储器子系统的机制(JCRE 没有提供直接访问的机制)。

javacardx.framework 程序包：该扩展包提供了一个框架，可以高效地实现典型的基于 JavaCard 技术的 Applet。该扩展包有三个子程序包，如果使用该扩展包必须全部包含这三个子程序包。

- util：提供了能够方便操作 short、int 和数组的函数。
- math：提供了存储数据、BCD 算法和奇偶校验的类。
- tlv：提供了创建和解析 TLV 对象和 TLV 结构数组的类。

作为通用智能卡操作系统，JavaCard 平台的通用性、兼容性和支持多应用的特性被广受好评，为许多业内人士喜爱。JavaCard 技术现在频频在智能卡不同领域中出现，相同的操作系统，截然不同的应用程序，JavaCard 犹如百变金刚，千变万化，神通广大。

12.1.3 出身名门——PC/SC 规范

1996 年 3 月，由 Microsoft、IBM、BULL 等八家 IC 卡厂商和 PC 软硬件厂商发起制定 PC/SC 规范。PC/SC 规范的全称是 Interoperability Specification for ICCs and Personal

Computer Systems，即智能卡与个人计算机系统互操作规范。

为了促进智能卡在 PC 环境中的应用，并使 PC 和智能卡之间的接口标准化，由 CP8 Transac（BULL）、Gemplus、Hewlett-Packard Company、IBM、Microsoft Corporation、Schlumberger SA、Siemens Nixdorf Information system AG、Sun Microsystems、Toshiba 和 VeriFone 联合成立了 PC/SC 工作组。

PC/SC 工作组成立的初始目的是为了发展一个规范，能够满足一种交互需要，使得 IC 卡技术能够方便地用于 PC 的工作环境下。除了发展这个规范，PC/SC 工作组成员们还致力于研究实现能够对这个规范有效的硬件和 PC 系统组件。

PC/SC 工作组的目标是：

（1）与现有的 IC 卡标准和 PC 的标准保持最大的一致性。

（2）建立不同运行平台中的各个组成部分之间的互操作性（与平台无关）。

（3）应用软件可以利用不同厂商提供的产品和组件（与厂商无关）。

（4）在不用重写应用层软件的前提下能够使用新技术（与应用软件无关）。

（5）增强 PC 上的基于 IC 卡应用软件的发展，促进 IC 卡服务的应用层接口标准化进程。

（6）建立一种在 PC 环境中鼓励使用 IC 卡的环境。

1. PC/SC 规范结构

本节介绍的 PC/SC 规范版本是 V 2.01.01，它有 10 个部分，共 11 份文件，规定了符合互操作性要求的设备、参考设计资料、编程接口和功能兼容性要求，包括：

PART 1：PC/SC 规范介绍与体系结构。

PART 2：IC 卡与接口设备的兼容需求。

PART 3：与 PC 互联的接口设备的需求。

PART 4：IFD 设计和参考信息，这一部分仅供参考。

PART 5：IC 卡资源管理器定义。

PART 6：IC 卡服务提供者接口定义。

PART 7：应用领域开发者设计需知。

PART 8：IC 卡设备的安全性与机密性实现建议。

PART 9：IFD 的扩展功能。

PART 10：IFD 的安全 PIN 登陆功能。

图 12-5 显示了本规范定义的硬件和软件结构，和规范的不同部分在系统中的位置。

2. PC/SC 体系结构

针对图 12-5 中 PC/SC 体系结构中的各部分进行详细描述。

（1）卡片（integrated circuit card，ICC）

本规范所规定的 IC 片应该在物理和电气特征上符合 ISO/IEC 7816-1、ISO/IEC 7816-2 和 ISO/IEC 7816-3 标准。本规范也支持符合 ISO/IEC 7816-10 草案的同步通信 IC 卡，还支持一些接触式同步和异步通信 IC 卡和非接触式 IC 卡，例如符合 ISO/IEC 14443 的非接触式近耦合卡（PICC）和符合 ISO/IEC 15693 的非接触式疏耦合卡（VICC）。

（2）接口设备（interface device，IFD）

本规范所规定的接口设备应该符合 ISO/IEC 7816-1，ISO/IEC 7816-2 和 ISO/IEC

7816-3 标准。另外，接口设备可以支持 ISO/IEC 7816-10 草案中规定的同步通信 IC 卡 1 和同步通信 IC 卡 2，或符合 ISO/IEC 14443 和 ISO/IEC 15693 的非接触式 IC 卡。设备与 PC 的接口可以是 RS-232、PS/2、USB、PCM/CIA 等。

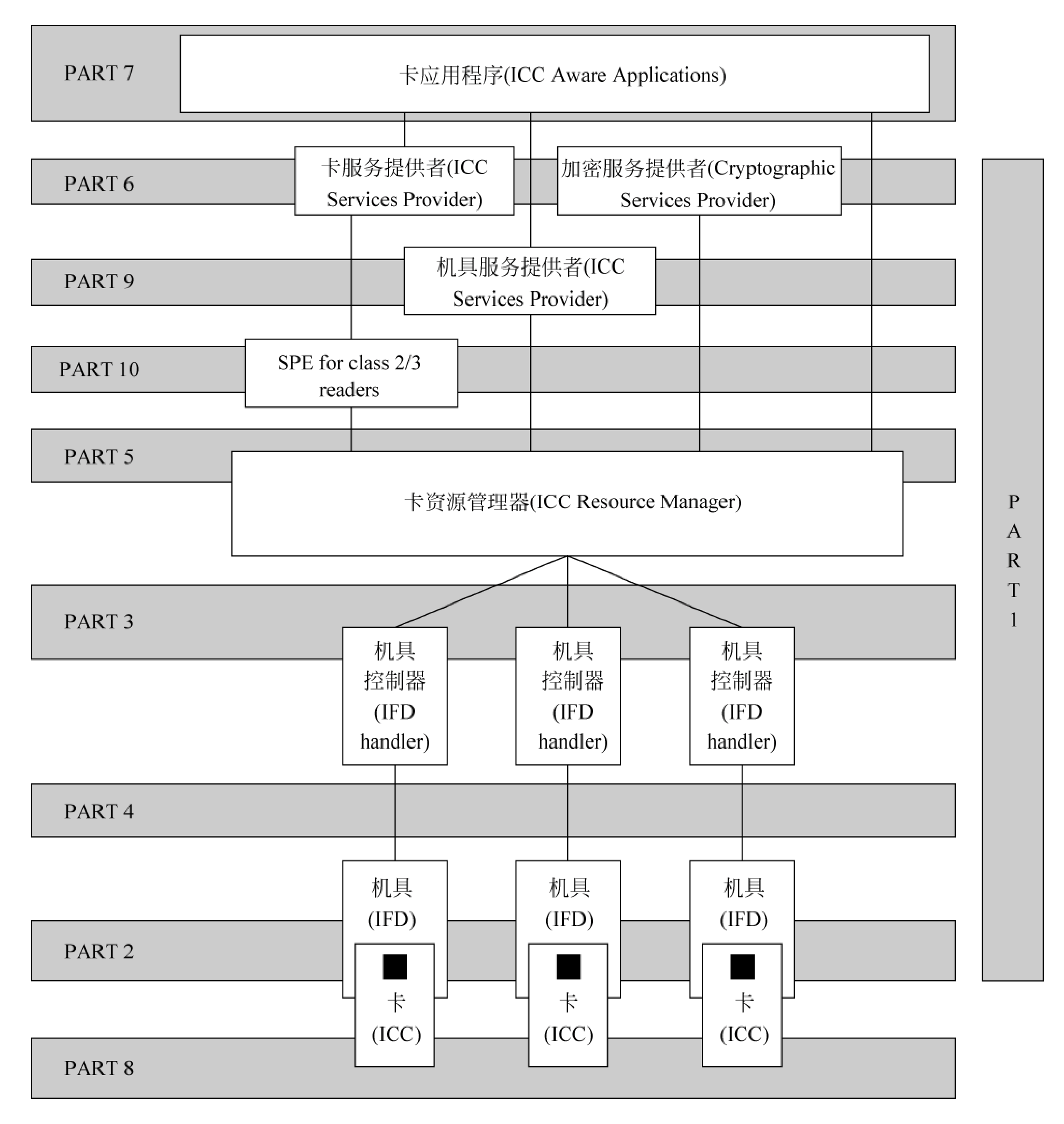


图 12-5　PC/SC 体系结构

(3) 接口设备处理程序(IFD handler)

接口设备处理程序是在 PC 上运行的底层软件，它的上层是操作系统中的卡资源管理器，下层是接口设备及其 I/O 驱动(如 PS/2 驱动、USB 驱动等，这部分在图 12-5 中并未标出)，所以说接口设备处理程序是将接口设备与 PC 连接的桥梁。它为上层应用程序提供了 API 函数，使上层应用的开发与不同厂商的接口设备和不同类型的接口无关。本规范并没有强制规定接口设备处理程序的实现形式，它可以是静态链接库、动态链接库或设备驱动。

接口设备处理程序可以支持两种类型的逻辑器件：槽逻辑器件(slot logical device)和

功能逻辑器件(functional logical device)。槽逻辑器件提供了与IC卡的通信链路。对于接触式接口设备,槽逻辑器件的数量与IC卡卡槽数量相同;对于非接触式接口设备,槽逻辑器件表现为与IC卡进行数据交互的信道。功能逻辑器件在原有的基础上提供了扩展功能接口,如显示屏、键盘等。

下面对接口设备处理程序完成的功能进行简要介绍。本规范将接口设备处理程序的功能分为必选功能和可选功能两部分。

必选功能主要有:

① IC卡事件管理:

- IC卡插入和拔出消息管理:如果IC卡插入和拔出事件发生,则接口设备处理程序会将此消息通知到上层的卡资源管理器。
- 系统功耗管理:为了降低系统功耗,接口设备处理程序可以将接口设备和卡片置于断电状态。

② IC卡接口管理:管理IC卡的激活和失效。对于接触式IC卡,激活功能完成与IC卡的连接和对ATR的解析;对于非接触式IC卡,激活功能完成IC卡的初始化与反碰撞。

③ 通信协议的支持:完成对通信协议(如ISO/IEC 7816/14443/15693)的物理层和链路层的支持。

④ 支持ISO/IEC 7816-4中规定的逻辑通道机制。

可选功能主要有:

① 支持存储卡相关命令操作。

② 支持机械动作,例如ATM机中卡片的吞入和吐出动作。

③ 支持其他安全设备,例如输入PIN码的键盘设备,对持卡人进行生物特征测量的设备等。

④ 支持接口设备厂商提供的其他特性。

(4) 卡资源管理器(ICC resource manager)

卡资源管理器是PC/SC体系结构中必不可少的系统级的一个组件,通常是由操作系统厂商提供的,负责管理系统中已安装的IC卡和接口设备。为管理多个IC卡和接口设备,卡资源管理器解决了3个基本问题。

首先,卡资源管理器负责资源的识别和管理,包括:

① 管理已安装的接口设备,并为其他应用程序提供这些信息。

② 管理已安装的IC卡、与IC卡相关联的服务提供者和提供的接口,并为其他应用程序提供这些信息。

③ 跟踪IC卡的插入和拔出事件,准确地维持接口设备中可用IC卡的信息。

其次,卡资源管理器负责控制接口设备被多个应用程序访问。为此,资源管理器提供了一种机制,在连接指定接口设备时,可以选择共享模式或独占模式。

最后,卡资源管理器支持处理简单的IC卡访问服务。这是非常重要的一个功能,例如当前的IC卡是单线程的设备,通常完成一个功能要执行多条命令,资源管理器可以处理多条命令不间断的执行,确保中间过程的完整性。

卡资源管理器是一个特权组件,控制了物理设备的访问并直接参与了卡应用程序(PC

上的)和IC卡之间所有命令和数据的交互。因此,确保不同进程的数据流之间的逻辑隔离是至关重要的。卡资源管理器被操作系统保护,维持在一个很高的安全级别,增加这个组件并不会对整个智能卡系统带来安全隐患。卡资源管理器的安装和配置需要有很高的安全特权,例如:用操作系统的管理员模式进行访问。

卡资源管理器应该提供用户接口(user interface,UI)来控制和管理IC卡和接口设备相关资源。用户可以通过UI实现以下功能:

- 安装或删除系统中的IC卡和接口设备。
- 列举已安装的IC卡和接口设备。
- 控制接口设备相关属性,如:用户可以自定义接口设备的名字和工作组、访问接口设备配置工具(由接口设备厂商提供)。
- 控制IC卡相关属性,如:用户可以自定义IC卡名字、通过ATR识别IC卡、与服务提供者相关联、与接口相关联。

UI还允许用户解决潜在的运行时资源冲突问题。UI并不是必须作为卡资源管理器的一部分实现,还可以通过UI模板或通用对话框的形式实现。

(5) 服务提供者(services provider)

服务提供者是建立在卡资源管理器之上的服务,将IC卡或接口设备的功能封装成API函数,提供给高层应用程序,使高层应用开发人员可以简化开发复杂度,不必过多关心IC卡和接口设备的全部技术细节。服务提供者包括卡服务提供者(ICCSP)、加密服务提供者(CSP)和接口设备服务提供者(IFDSP)。

ICCSP负责为高层应用接口提供卡相关的非加密服务,包括IC卡的连接管理、文件访问和认证服务。

文件访问服务可以完成以下功能:通过文件名定位文件;创建或打开文件;文件内容的读写;关闭文件;删除文件;文件属性管理。

认证服务完成以下功能:持卡人身份认证;IC卡认证;应用程序认证。

CSP负责为高层应用接口提供卡相关的加密服务,因为不同国家政府对信息安全产业的政策不同,所以服务提供者将CSP从卡服务中提取出来,以便适应不同国家的加密服务需求。

CSP服务可以完成以下功能:密钥的产生;密钥的管理;数字签名;散列运算(或消息摘要);大量的加密服务;密钥的导入和导出。

ICCSP和CSP还可以提供UI,这是可选的。UI可以提供以下服务:持卡人身份鉴权;口令和PIN码管理;管理访问重置或禁用CHV功能。

IFDSP是一个可选的组件,由接口设备厂商提供,通过IC卡资源管理器与接口设备处理程序通信。IFDSP可以为高层应用程序提供不同类型的接口服务,例如PIN保护、显示、用户确认和普通用户入口。

(6) 卡应用程序(ICC-aware applications)

卡应用程序是针对不同的智能卡应用环境以及需求而编写的软件程序。本规范定义了卡与PC应用程序之间的映射机制——单用户、单线程、多应用。

应用程序开发人员通常利用服务提供者和卡资源管理器所提供的资源编写卡应用程序,可以通过API函数,或用C++、Java和ActiveX编写的接口模块访问上述资源。如果需

要，还可以自行开发 ICCSP。

虽然本规范与平台无关，但是仍然建议操作系统可以为应用程序提供以下服务：

- 多线程处理。多数应用程序至少需要两个线程，一个线程用来监听卡插入和拔出的事件，另一个线程用来处理和卡之间的通信、UI 的消息等。
- 异步事件和消息处理。用来侦测卡的插入和拔出。
- 可以动态链接分享代码的共享数据库机制。当卡插入接口设备时可以装载相应的 ICCSP。

应用程序开发人员至少需要掌握 PS/SC 的 PART 5/6/7 三部分，在 PART 7 中详细介绍了应用程序开发需知，主要讨论了以下几种任务的设计方法：

- 动态配置可用系统资源。
- 连接指定的 IC 卡。
- 设备共享。
- 卡片的安全服务。
- 错误校正。

PC/SC 规范是由微软大力推行的，由于目前采用 Windows 操作系统的计算机绝对占据主流，所以 PC/SC 规范拥有良好的发展前景，甚至还有在 Linux 等操作系统下移植 PC/SC 的案例，可见出身名门的 PC/SC 影响力果然不凡。

12.1.4 后起之秀——ISO/IEC 24727 规范

ISO/IEC 7816 和 ISO/IEC 14443 仅仅规定了卡片本身的底层接口和规范，由于规范的复杂性和专业性，目前对智能卡的开发需要专业的知识和大量的时间才能完成，并且开发出的系统基本上都是一个个独立的系统，只能是在系统内适用，或者本行业内通用，与其他系统之间不具有兼容性，这在很大的程度上制约了智能卡的发展。

为了解决此问题，ISO/IEC 24727 应运而生，它定义了智能卡和外部应用之间互相交互的编程接口。不论卡片的类型和接口是接触式或是非接触式，只要是面向客户的智能卡应用，该标准就为应用程序和中间件，定义了一个高级的接口和统一的规范。该标准可以有效地预防客户定制方案的出现，并且屏蔽来自不同厂商的特定应用之间的不同。接下来对该标准进行简单的描述。目前所存在的系统如下图 12-6 所示，每个系统都是一个孤立的信息孤岛，互相之间没有通用性。

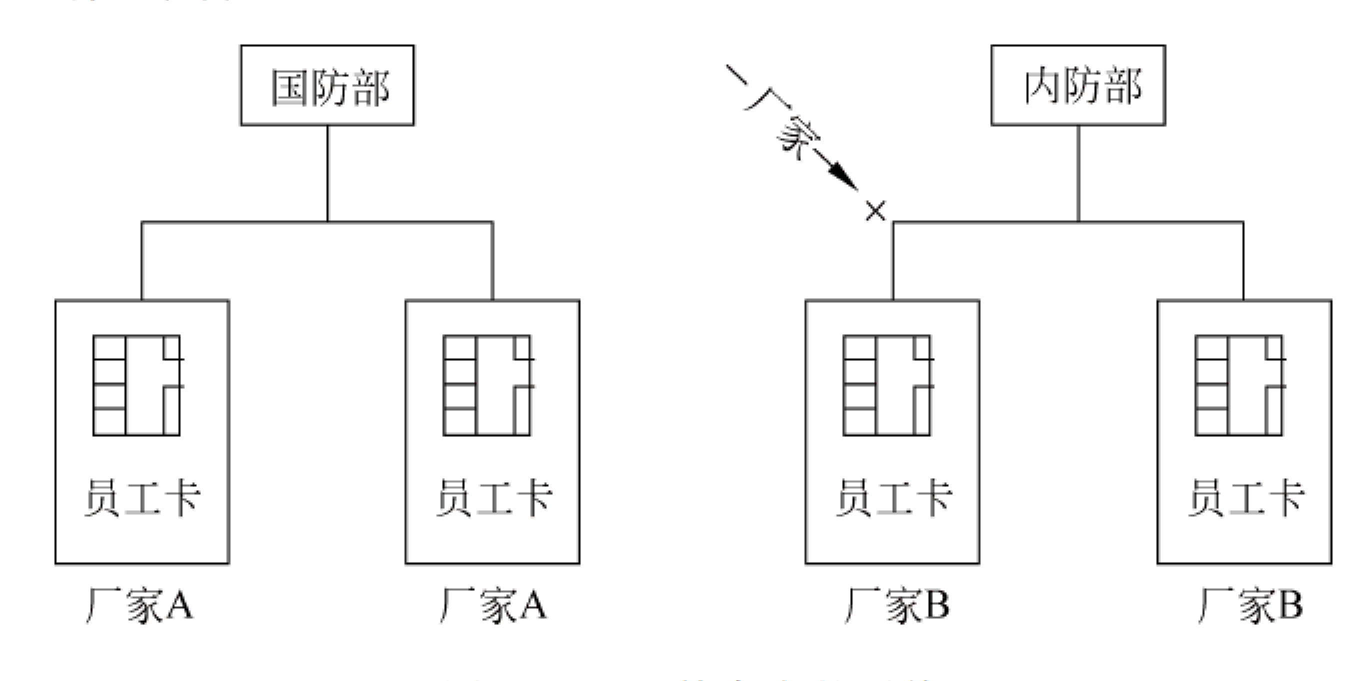


图 12-6 目前存在的系统

在系统中添加了中间件后各个系统之间的卡片可以互相通用，情况如图 12-7 所示。

图 12-7　添加中间件后的系统

只要卡片符合 ISO/IEC 24727 规范，就可以在世界上任何一个符合此规范的应用系统中使用，这将极大地方便人们的生活，将成为智能卡未来发展的方向，如图 12-8 所示。

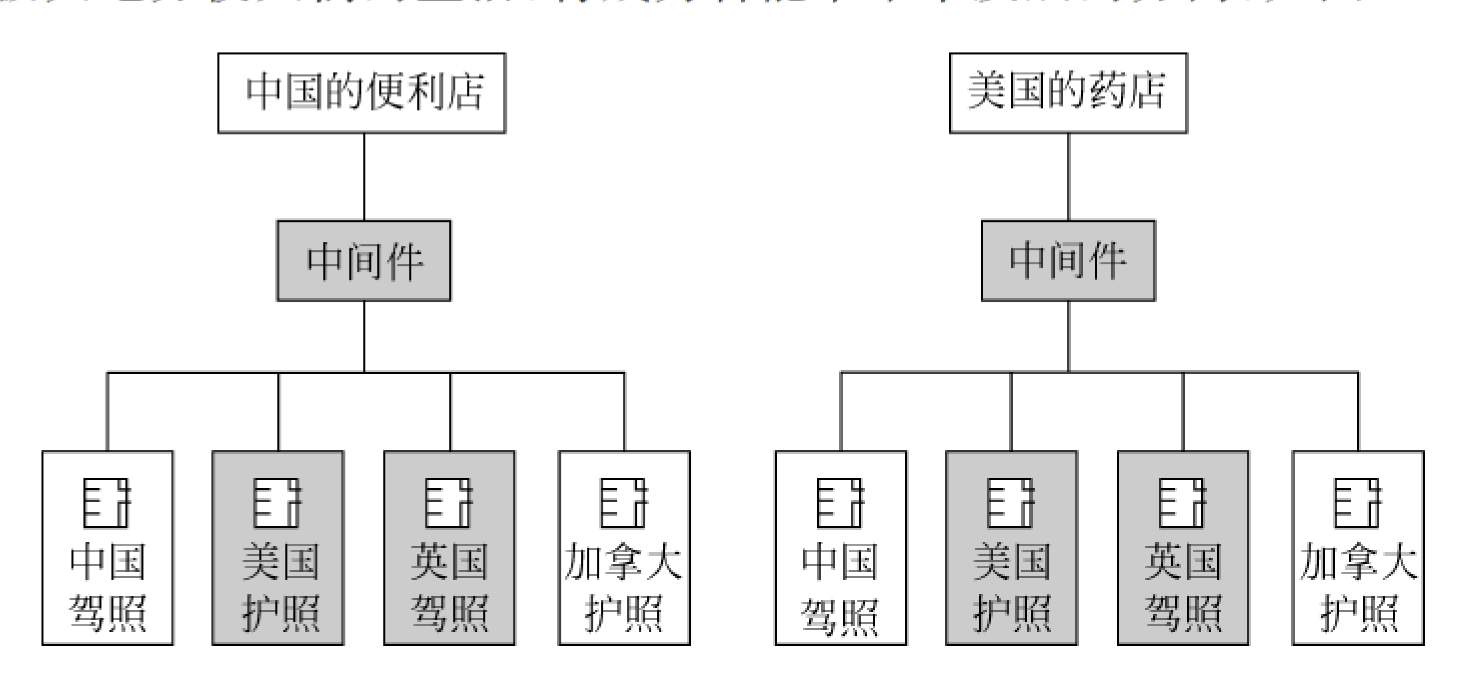


图 12-8　符合规范的卡片应用场景

随着普通消费者对智能卡兼容性要求的日益迫切，以及各个厂商和应用方对兼容性卡片获取利润的与日俱增，未来智能卡中间件有望统一在后起之秀 ISO/IEC 24727 标准的麾下。

更多技术细节可参见 ISO/IEC 24727 标准。

12.2　卡未来无限憧憬

12.2.1　天马行空——性能提高

随着半导体技术的发展和智能卡日趋复杂的应用需求，智能卡芯片的性能也不断提升。CEA Leti、Gemalto 和 Raisonance 联合开发了基于非接触式智能卡的 VHDR(very high data rate)技术和协议，通信速率最高可达 10Mbps，比现在的通信速率快了 10 倍有余(现在 ISO/IEC 14443 规定的通信速率最高为 848kbps)，非常适合未来高数据量的智能卡应用，例如电子护照的大量数据认证和存储医学图像(如 X 光图片、CT 照片等)的医疗卡。VHDR 与现在的 ISO/IEC 14443 通信协议兼容，而且具有极高的安全性。在 2009 年巴黎卡展上，CEA Leti、Gemalto 和 Raisonance 展示的 VHDR 技术通信速率达到了 6.7Mbps，根据图片的大小不同，传输一幅高分辨率的医学图片仅需 0.5～5 秒，在不久的将来速率会

提升到 10Mbps。VHDR 技术获得了该年度的巴黎芝麻大奖的最佳硬件奖。CEA Leti、Gemalto 和 Raisonance 的非接触 VHDR 如图 12-9 所示。

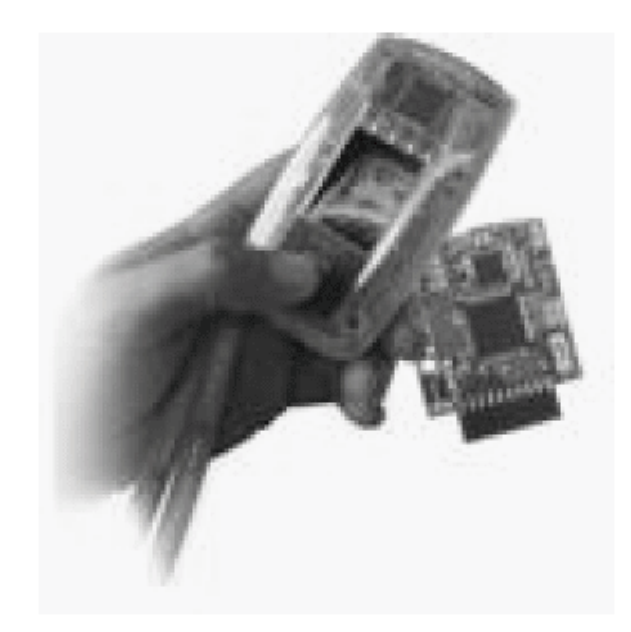


图 12-9 CEA Leti、Gemalto 和 Raisonance 的非接触 VHDR

当前市场上主流的智能卡 CPU 是 8/16 位处理器，独立研究表明，由于现有 8/16 位传统解决方案难以满足新一代智能卡在功能、性能及能效方面的需求，32 位安全处理器市场正在取得显著的增长。ARM 公司在 2008 年宣布启动了 ARM SecurCore 代工厂计划，我国的同方微电子和华大电子以及代工厂华虹成为该计划的首批合作伙伴。在 2010 年香港举办的亚洲智能卡工业大会上 ARM 公司宣布推出高度紧凑、节能型 ARM®SecurCore™SC000™处理器，该款处理器专为最高容量的智能卡和嵌入式安全应用而设计。SC000 能够以具有竞争力的 8/16 位成本、面积和功耗提供前所未有的丰富的 32 位性能，是最高容量智能卡市场下一代设备的理想之选。ARM 公司的 SecurCore 系列处理器已被多家智能卡芯片供应商授权获得，包括 Atmel、NXP、Samsung、ST 和 Toshiba 等。

除了非接触通信速率和处理器位数的提升，智能卡的容量也在不断增加。SK 电讯在巴塞罗那举办的 2010 年移动通信世界大会上对外宣布其研发的高性能 Smart SIM 卡将于本年 5 月在全球领先投入商用。Smart SIM 是安装于 3G 手机的 USIM 卡，卡内搭载高性能 ARM9 处理器，内存高达 1GB，使卡片不仅可以储存电话簿、短消息、游戏、多媒体资源等丰富内容，还可以安装包括 Android 在内的多种终端机应用程序，使用户即使更换手机，也可以保留原来的用户界面环境并继续使用原来的应用程序内容。今后，移动运营商和应用提供商可以基于 Smart SIM 开发出不受终端机型限制的新型增值服务，而终端制造商则可以减少对移动运营商要求的增值服务和程序开发，从而加大对终端机的设计与质量上的投入力度，节省开发成本和时间。

上述案例都是当前可以预期的最新技术，由此可见，智能卡性能的不断提高，为我们天马行空般地想象未来的智能卡留下了极大的空间。试想，拥有高性能处理器、高容量内存、高速率数据交互、集便携性和安全性于一身的智能卡，配上友好的人机操作接口将是什么，掌上电脑？PDA？现在有电信网、电视网和因特网的三网融合，那么未来智能卡会和掌上电脑、PDA 甚至手机融合吗？图 12-10 形象地说明了对智能卡未来的畅想。

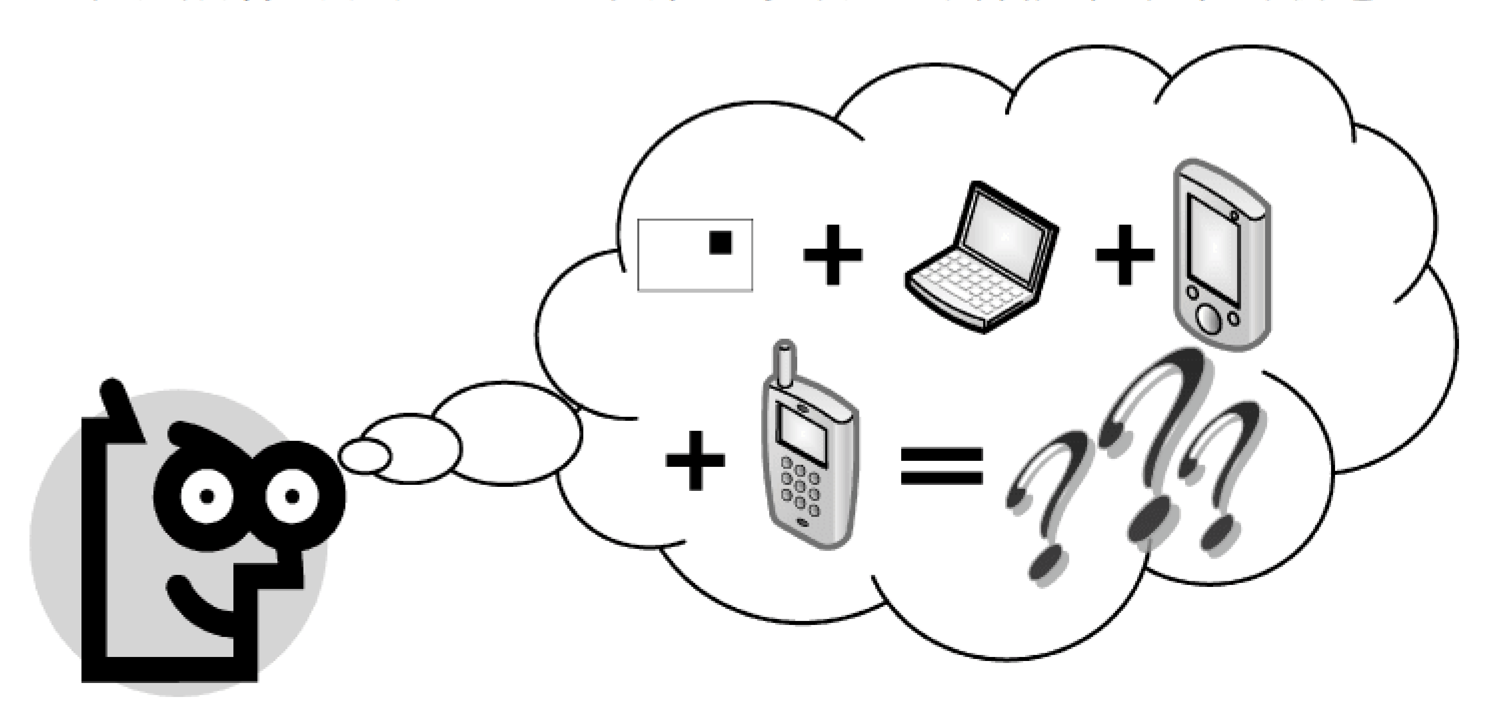


图 12-10 畅想卡未来

12.2.2 独具匠心——技术融合

如果12.2.1节所说的智能卡与掌上电脑、PDA甚至手机的"合体"有些痴人说梦的话，那么至少从现在看来，智能卡技术与其他技术，如生物特征、射频识别、信息安全、Web、材料、光学防伪等技术的继续融合是未来智能卡发展的重要趋势。

2009年巴黎卡展所评选的10个芝麻大奖中，3个获奖设计都是将已有的智能卡技术和其他技术结合。

GEMATIK公司的eHealth-Portal解决方案是将智能卡插入到客户端的读卡器中，无需任何驱动即可将卡中的信息与远程数据中心进行认证和交互。eHealth是集远程医疗、健康咨询、健康宣教为一体的多功能应用服务平台。德国的eHealth项目包含了8千万患者、35万医生、2000家医院和22000家药店，被誉为目前世界上最大的eHealth工程。我国在上海、江苏等地也有eHealth试点。eHealth平台示意图如图12-11所示，GEMATIK公司的eHealth-Portal完美地解决了居民端的数据接入和共享问题，居民可以将病例、医疗记录等信息存入智能卡，通过GEMATIK公司提供的设备即可通过网络进行卫生保健服务。此项技术获得了最佳医疗应用奖。



图12-11 GEMATIK公司的eHealth-Portal

JDSU公司的HoloFuse是一种新的聚碳酸酯薄膜，无需粘合剂即可将全息图案(holographic pattern)"粘"在其表面，如图12-12所示。聚碳酸酯是近年来证件类常用的一种材料，全球范围内大约有40%的护照和大部分国家的身份证都是采用这种材料制成的；

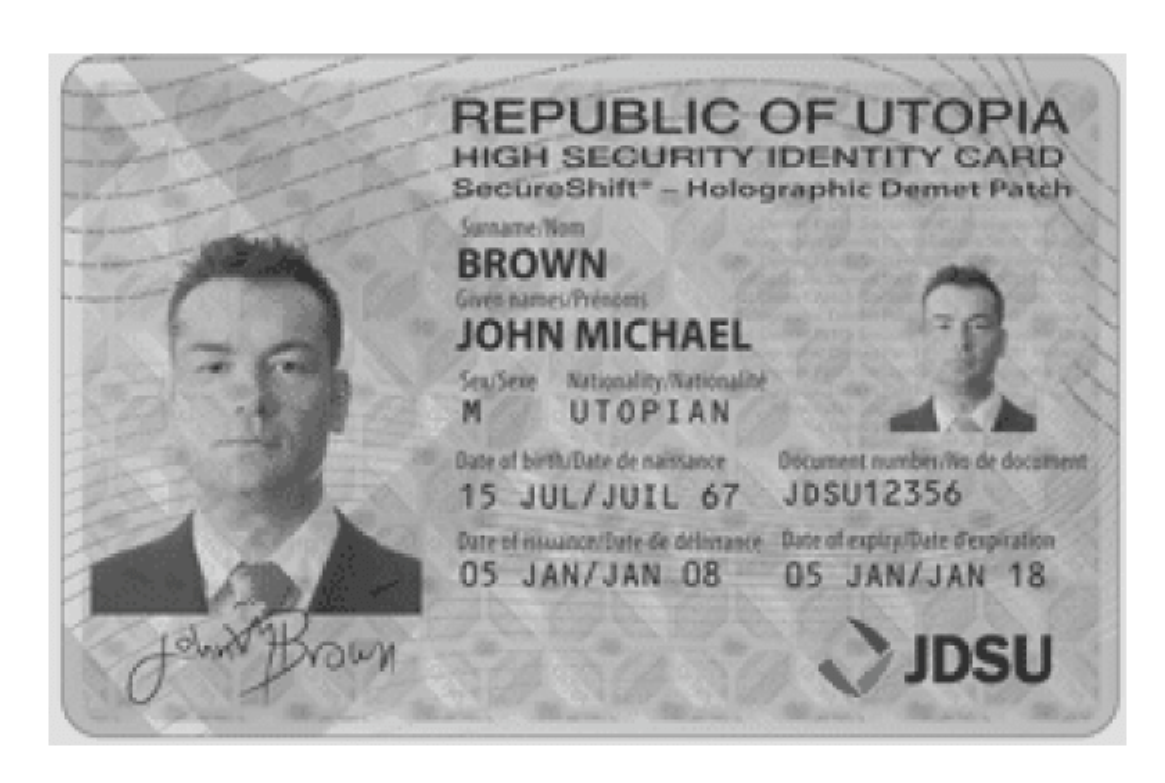


图12-12 JDSU公司的HoloFuse

全息技术是一种有效的可视防伪技术，被ICAO在护照中推荐使用，而且欧盟已经强制其成员国在护照中结合此技术。将聚碳酸酯和全息图案结合的传统方法有两个缺点：第一是图案的亮度不但会因为制作过程中的加热和压力而变暗，而且随着时间的推移图像的色彩也会变得模糊；另一个是聚碳酸酯层和全息图案层可能会被攻击者使用某种手段分离。HoloFuse不仅克服了这两个缺点，而且和激光雕刻技术兼容，可以很轻松地通过全息图案和激光雕刻对个人数据进行有效的防伪。HoloFuse技术瞄向了身份证件安全市场，例如身份证、护照、驾照等。此项技术获得了最佳身份认证应用奖。

MONEXT公司的Sign4Pay(外形如图12-13所示)是一种利用手机的在线交易安全系统，是结合了WPKI服务和基础设施的新型在线支付方式，这种解决方案可以将SIM卡和电子身份相关联，使用户在线交易变得更安全、操作更简单。这个设计是基于一个非常简单的道理：如果你想避免别人偷走你的信用卡号码，那就别使用它！关于移动PKI的实验越来越多，但是到目前为止都没有在电子支付中应用。MONEXT提议使用手机进行电子支付，这样就拥有了两个强大的认证实体——银行和电信运营商。当在手机中进行在线购买产品时，会收到手机交易的请求，只需输入预先设定的PIN码，就大功告成了，认证、转账和邮件交易三步操作无缝连接。手机成为了一个认证/身份证明，目的是完全以用户为中心，简化在线支付流程。MONEXT公司的Sign4Pay获得了最佳电子支付应用奖。



图12-13　MONEXT公司的Sign4Pay

智能卡是一门综合性很强的技术，具有高度的灵活性，根据实际的需要，充分动用开发人员的想象力，与其他一种或几种技术相互借鉴，就可能产生出一种新的独具匠心的技术方案或解决方案。

12.2.3　七十二变——一卡多用

有没有想过有一天你的钱包被身份证、驾照、机动车行驶证、加油卡、银行卡、信用卡、公交一卡、校园卡、医保卡、社保卡等十几种卡片塞得鼓鼓囊囊？有没有遇到过从十几种卡片中费尽周折地找到想要用的卡片？

单应用智能卡具有管理携带相对复杂、卡中数据重复、资源利用率低和添加应用或增值服务困难的缺陷；而多应用智能卡可以提高智能卡的资源效率，降低卡的使用成本，减少持卡人携卡数量，充分体现智能卡便捷的特性。

多应用卡的技术核心问题是：

- 卡上多个应用程序的彼此独立和数据（文件）的隔离与共享。
- 终端与卡上多应用程序的相互识别和认证。
- 卡发行商或服务提供商对各自应用的管理（安装、更新、删除等）。

虽然依靠现有的技术解决上述核心问题并不十分困难，例如：开放式操作系统，如JavaCard平台拥有应用防火墙和数据共享机制（详情请参考12.1.2节）；GP规范定义了应用的下载方法和各卡外实体对卡内容管理的权力与义务（详情请参考7.5节及12.1.1节）。但是多应用智能卡的安全问题依然值得重点关注，因为一旦应用程序的下层（如操作系统COS或智能卡硬件）发现重大安全漏洞，那么卡上的所有应用程序的安全性都会受到威胁，带来的损失会比单应用智能卡更加恐怖！

尽管目前市场上单应用智能卡仍占据主流，但多应用卡所占的比重正逐年增多，涌现了各式各样的多应用卡的解决方案。

握奇公司在新加坡推出的LiveFreshVisa铂金信用卡就是在亚太地区首次在一张智能卡上融合了交通支付、信用卡以及非接触小额支付的功能。这极大地方便了人们的日常生活，持卡人可以在便利店用电子钱包消费，可以在商场刷卡信用消费，还可以在出行时刷卡乘车。这个产品获得了2010年首届亚洲智能卡展的"亚洲芝麻奖"。

一卡多应用这个概念出现的时间已经不短，但是现在市场上多应用卡并没有占主导地位，而且不同应用的融合程度不高，最主要的原因并非技术问题，而是政策与利益问题。

政策问题：以公交IC卡为例，其属于在本行业内采用新技术手段实现行业收费的应用，本身不存在任何政策问题。但如果拓展到其他行业进行跨行业收费应用，则形成在较大范围流通、涉及金额较大、具有类似金融支付能力的电子现金系统，处于"灰色区域"，面临国家相关金融法规和政策是否允许的问题。

利益问题：从单一行业应用走向多行业应用，面临的最主要问题之一就是利益问题。任何行业的既得利益者对新的进入者都不会欢迎，行业实际上就是利益集团的代名词，是该行业应用者之间的一种利益联盟。所谓行业准入或行业壁垒，就是该行业的既得利益者，为了防止新的进入者导致利益格局的再分配，从而设置了一系列的障碍和门槛，这也是行业之所以形成的基础。

行业标准实际上就是利益的最高级的实现手段。国际标准主要有两种类型：一种是事实标准，即行业领导者在该行业内已经具有绝对领导和支配能力、基于既成事实而其他后来进入者不得不承认或接受的标准；另一种是定制标准，是由该行业的部分领导者主导、发起制定的、具有利己排他性（如：根据自己的专利技术等定制）优势和领先能力的标准。

尽管有重重困难，但随着智能卡技术和市场需求的发展，一卡多用会逐渐成为主流。智能卡的硬件、COS和应用程序也将会越来越复杂，根据应用场合的不同，嵌入式的差异可能会越来越明显，犹如孙悟空的七十二般变化，外形迥异，看得人眼花缭乱。原有的安全评定方法所设的等级和条目可能有些简单，无法满足日益复杂的智能卡，所以未来安全评价方式可能也会发生变化，甚至借鉴计算机安全等级保护来评价智能卡的安全等级。

12.2.4 随心所欲——移动办公

“移动办公”也可称为“3A办公”，即办公人员可在任何时间（anytime）、任何地点（anywhere）处理与业务相关的任何事情（anything）。这种全新的办公模式，可以让员工办公摆脱时间和空间的束缚，随心所欲、随时随地地进行信息的交互流动。

移动办公的发展与移动设备和通信技术的发展是密切相关的，移动办公的发展经历了三个阶段：离线式移动办公、有线移动办公、无线移动办公。离线式移动办公出现于20世纪90年代，笔记本电脑、PDA等设备的广泛应用为这个阶段移动办公的发展提供了契机，人们可以带着笔记本电脑到几乎任何地方进行工作，这让移动办公成为了可能，但由于当时的因特网技术还没有广泛普及，信息的交换主要通过回到办公室以后同步才能完成。有线移动办公是随着因特网技术的发展而出现的，由于VPN技术的出现，人们可以借助VPN提供的安全通道接入有线网络，从而实现有线的移动办公。无线移动办公是以CDMA和GPRS移动通信技术的发展为主要特征的，随着移动通信技术的迅猛发展，智能手机的广泛普及，移动通信已经从2G跨入了3G时代，为移动办公提供了更加先进的移动通信平台。

安全问题始终是移动办公首先要解决的问题。由于便携式电子设备在移动办公中的广泛应用，设备的失窃事件时有发生；又如计算机病毒、木马、网络黑客的出现，数据失窃，机密文件被盗，系统遭到攻击等事件频频出现；手机病毒的出现对无线移动办公提出了挑战。移动办公是否能够解决安全问题，是移动办公技术能否广泛应用的关键问题。目前在移动办公领域主要应用的信息安全技术有防火墙、加解密、数字签名、VPN技术虚拟专用网等。这些技术较为成熟，为无线办公提供了较好的安全防护策略。此外PKI技术提供的机密性、真实性、完整性、不可否认性为移动办公提供了很好的解决方案。同时，作为信息安全中重要应用，智能卡具有携带方便和数据安全的特点，提供了身份认证和多种安全机制，由于其天生具有的便携性和安全性，智能卡技术在移动办公技术中的应用，在很大程度上提高了移动办公系统的安全性。智能卡技术为通向移动办公的大门加上了一把安全的锁，而智能卡则是打开这个大门的钥匙。

虚拟机技术的快速发展，为开发跨平台应用系统提供了契机。Java技术则是虚拟机技术的实际应用，Java具有很好的平台无关性，通过在计算机上构建虚拟机，应用程序可以实现“一次编译，随处运行”。作为虚拟机技术的领跑者，Sun公司很早就把这种平台无关的思想贯彻到移动办公应用之中，Sun公司的每名员工都有一张代表自己身份的JavaCard，名片大小的卡片记录着包括使用者身份、个性等在内的所有信息。Sun的员工无论在全球哪个办公室，都只用随身携带JavaCard，只要在带有读卡器的终端客户机上一刷，就可以调出自己所有的文档，甚至连桌面都会与自己上次关机时一模一样。Sun公司推出的Sun Ray系统则是上述系统的实现，它是一款定位在集成应用和状态无关的瘦客户端解决方案，它的发展已经有了近十年的历史，它的特点在于引入的智能卡技术。状态无关桌面系统的思想是从Sun早期的无磁盘Java桌面系统发展而来。SunRay客户端是一个显示设备，而应用程序在服务器上运行，用户的会话状态和显示无关。这就使SunRay操作系统具有了移动的特点：用户能够很容易地把工作从一台SunRay转移到另一台SunRay上。而用户需要做的仅仅是，刷一下卡，输入密码，然后就可以在新的SunRay上继续工作。

3G时代的到来为移动办公的发展提供了新的发展机会，3G通信的最大特点是带宽的

增加和速率的提升，这使得原本在2G时代对网络传输速率要求较高的业务如视频会议、电子邮件实时收发、可视电话、多媒体通信等成为可能。智能手机的出现使得手机办公得以迅猛发展，现在很多智能手机都集成了对word、ppt、pdf等文档的编辑功能，能在手机上轻松的收发电子邮件和访问网络。USIM卡正从单纯的认证功能向移动商务平台，乃至多应用平台过渡，可以在手机上实现电子钱包、电子信用卡、电子票据等多种应用。此外USIM卡还在安全性上对算法进行了升级，增加了卡对网络的认证功能，这种双向认证可以有效地防止黑客攻击。USIM卡在3G领域的应用为包括移动办公应用在内的多应用领域打开了一扇大门，随着技术的不断进步，基于USIM卡的多应用终会在3G时代得到广泛应用，移动办公业务也可以广泛的普及。

随着智能卡的通信速度，存储容量和安全性等方面性能的提高，智能卡在移动办公领域的应用会更加广泛。智能卡的应用将不仅局限于身份认证，还将包括机密数据的存储、安全机制的实施、多应用等领域。智能卡技术将为移动办公提供可靠的安全保障和移动便携解决方案，我们有理由期待移动办公的未来更加美好。

12.3 物联网风起云涌

12.3.1 连接万物——未来物联网的发展蓝图

什么是物联网呢？这一切先要从国际电信联盟(International Telecommunication Union，ITU)在2005年发布的一篇报告说起。

《ITU因特网报告2005：物联网》(*ITU Internet Report 2005：The Internet Of Things*)是ITU在2005年Tunis举办的信息社会世界峰会上发布的报告。该报告全面而透彻地分析了物联网，为世人第一次揭开了物联网神秘的面纱。

当今，全域信息和通信网络技术的迅猛发展，使信息技术和通信技术的世界加入了新的维度，从过去的任何时间(anytime)、任何地点(anywhere)连通任何人(anyone)，增加了连通任何物体(anything)。连接的增加会创建一个全新的动态网络——物联网。

报告并没有明确给出物联网的参考定义，但是用了一个很经典的通信技术描述词语——anytime、anywhere、anyone和anything以及关键字connect为我们诠释了物联网的含义。报告称物联网是建立在无限传感技术和纳米技术等重要领域的技术创新基础之上，预示着计算机和通信技术未来的一次技术革命。首先，为了连接日常用品和设备并导入到大型数据库和网络，一套简单易用并有效的物体识别系统是至关重要的。只有这样，物体的数据才能够被收集和处理。RFID技术就提供了这样的功能。其次，采集物体数据时还需要有描述物体属性的数据，可以使用传感器技术来探测物体的状态与属性。然后，物体中的嵌入式智能技术(例如智能家庭、智能汽车、私人机器人等)可以通过将信息处理转移到在网络边界从而增强网络的能力。最后，小型化技术和纳米技术可以使体积微小的物体交互和连接。把这些技术融合到一起，就形成了物联网，将世界上的物体从感官上和智能上连接到一起。

物联网可以给消费者、制造商和各类企业都带来巨大的潜力，但是报告也称未来物联网的商业化运作会十分艰难，标准的统一更是难上加难。报告把RFID技术、传感器技术、智

能技术和小型化及纳米技术列为实现物联网的四个基础技术。RFID 技术是其中最成熟的一个,已经建立了标准协议并拥有成熟可观的商业运作规模,EPC Global 也在推进 RFID 标准化工作;无线传感器网络被广泛使用在汽车、家庭安保、医疗、航天、远程监控、地质环境监测等行业,标准化工作正在被 ZigBee 联盟和一些其他组织推进;而小型化及纳米技术则因为缺乏共识和协商而遥遥无期。

数据的安全和隐私的保护是物联网面临的重要挑战。由于日常物品上贴有物品识别的标签并能被传感器感知,加上强大的信息处理、计算与传输功能,我们的日常生活将被成千上万的“眼睛”、“耳朵”和“鼻子”包围,谁将拥有管理这些“五官”的最高权限呢?

报告称物联网不仅是工业大国的私人领地,还是发展中国家的机遇,为他们在药品制作、水源洁净、能源产业、食品安全等方面带来许多应用。物联网的发展将会伴随一个由各产业巨头驱动的新生态系统的诞生。这些巨头必须操作出一系列持续发展的经济和法律系统,这样就能为他们的最终盈利提供框架。但是,人本身依旧是整个生态系统的核心,因为人类的需求对未来的领域创新的影响才是关键。事实上,技术和市场不可能脱离社会和伦理体系而独立存在。物联网将从很多方面改变我们的日常生活,影响我们的行为,甚至价值观。

报告最后还讲了一个小故事,假想 2020 年西班牙女学生 Rosa 的一天,从故事中我们可以形象地想象物联网将为我们的生活带来怎样的变化。作者将这个小故事稍作修改,主角是北京 24 岁的女学生小雅。

小雅刚刚结束和男友的争吵,需要一段时间自己一个人静一静。她打算开自己的智能红旗轿车到南戴河的一个著名的海滩浴场度过周末。但是好像她需要在汽修厂停留一会儿。她的爱车依法安装的 RFID 传感器在警告可能的轮胎故障。当她经过她喜爱的汽修厂入口处的时候,使用无线传感技术和无线传输技术的诊断工具对她的汽车进行了检查,并要求其驶向指定的维修台。这个维修台是由全自动的机器臂装备的。小雅离开自己的爱车去喝点咖啡。智能饮料机知道小雅喜欢喝冰咖啡,当她利用自己的手机钱包安全付款之后立刻倒出饮料。等小雅喝完咖啡回到修车场时,一对崭新的轮胎已经安装完毕,并且集成了 RFID 标记以便检测压力、温度和形变。

这时机器向导要求小雅注意轮胎的隐私相关选项。汽车控制系统里存储的信息本来是为汽车维护准备的,但是在有 RFID 阅读器的地方,旅程的线路也能被阅读。因为小雅不希望任何人(尤其是男友)知道自己的动向,这样的信息太敏感了,所以她选择隐私保护来防止未授权的追踪。

在高速公路收费站,小雅没有停车,因为她的汽车里包含了她的驾照信息和收费卡,已经自动传送到相关系统了。

忽然小雅在自己的太阳镜上接到了一个视频电话请求。她选择了接听,看到她男友正在请求她的原谅,询问她是否愿意共度周末。她喜从中来,马上发布指令要求导航系统禁用隐私保护,这样男友就能找到她的位置直接过来了。

瞧!即使是在这样一个充斥着智能互联系统的世界,人类情感还依然是主宰!

12.3.2 群雄割据——各国物联网发展战略概述

物联网是未来信息技术的一次技术革命,可以刺激国家的经济发展,世界各发达大国的

IT 产业巨头,甚至国家领导人都对物联网十分关注。各国在相互合作、共同促进全球物联网标准化的同时,也都在加紧研发关键技术,抢占制高点,企图拥有更多的国际话语权。虽然各国技术基础有差距,但是现在物联网的研发工作起步时间比较短,差距还不明显,所以用群雄割据来形容目前各国的物联网发展的现状毫不为过。下面我们将对美国、欧洲、日本、韩国以及我国的物联网战略做一下简要介绍。

首先介绍美国的物联网发展战略。IT 巨头 IBM 在 2008 年底率先提出了“智慧地球”的概念,IBM 认为 IT 产业下一阶段的任务是把新一代 IT 技术充分运用在各行各业之中,具体地说,就是把感应器嵌入到电网、铁路、桥梁、隧道、公路、建筑、供水系统、大坝、油气管道等各种物体中,并且被普遍连接,形成所谓“物联网”,然后将“物联网”与现有的因特网整合起来,实现人类社会与物理系统的整合,在这个整合的网络当中,存在能力超级强大的中心计算机群,能够对整合网络内的人员、机器、设备和基础设施实施实时的管理和控制,在此基础上,人类可以以更加精细和动态的方式管理生产和生活,达到“智慧”状态,提高资源利用率和生产力水平,改善人与自然间的关系。2009 年 1 月 28 日,在奥巴马就任总统后的第一次美国工商业领袖圆桌会议上,IBM 首席执行官向奥巴马进言,建议新政府投资新一代的智慧型基础设施。此举得到了奥巴马的积极回应,将“智慧地球”提升到国家战略的高度,并在随后出台的《经济复苏与再投资法》中对这一战略加以落实。

其次介绍欧洲的物联网发展战略。欧洲智能系统集成技术平台(EPoSS)在 2008 年发表了一篇报告 *Internet of Things in* 2020,详细地讲述了未来物联网在技术、应用方面的发展趋势,并预测了未来十几年内物联网的四个发展阶段(这部分内容会在后面章节简介)。2009 年,欧盟执委会发表了题为 *Internet of Things -An action plan for Europe* 的物联网行动方案。该方案描绘了物联网技术应用的前景,并提出要加强欧盟政府对物联网的管理等 12 项行动来保障物联网发展。

然后介绍我们的近邻日本和韩国的物联网发展战略。进入 21 世纪以来,日本积极推进 IT 立国战略,取得了令人瞩目的成就。政府相继制定了 e-Japan、u-Japan、i-Japan 等多项国家信息技术发展战略,从大规模开展信息基础设施建设入手,稳步推进,不断拓展和深化信息技术的应用,以此带动本国社会和经济发展,其中 u-Japan、i-Japan 战略与目前物联网概念有许多共通之处。最新的 i-Japan 战略是在 2009 年 7 月提出的,目标是让数字信息技术融入每一个角落。韩国是一个大力支持民族工业的国家,自 1997 年起韩国政府就出台了一系列推动国家信息化建设的产业政策,并取得了相当的成绩,在此基础上,2009 年韩国通信委员会出台了《物联网基础设施构建基本规划》,将物联网市场确定为新增长动力。计划提出到 2012 年实现“通过构建世界最先进的物联网基础实施,打造未来广播通信融合领域超一流信息通信技术强国”的目标,围绕此目标,已经部署了一系列课题的研究。

最后介绍我国的物联网发展战略。在美国、欧洲、日韩相继把物联网上升为国家发展战略时,我国也毫不落后,国家领导对物联网高度关注。2009 年 8 月 7 日,温家宝总理在中科院无锡高新微纳传感网工程技术研发中心考察时指出,要积极创造条件,在进入微纳传感领域比较早的无锡市建立中国的传感网中心(“感知中国”中心),早一点谋划未来,早一点攻破核心技术,抢占传感网技术和产业制高点。同年 11 月 4 日,温总理在人民大会堂向首都科技界发表了题为《让科技引领中国可持续发展》的讲话,其中指出要着力突破传感网、物联网关键技术,及早部署后 IP 时代相关技术研发,使信息网络产业成为推动产业升级、迈向信息

社会的“发动机”。移动通信是物联网中信息传输的重要环节，2009 年 11 月，中国移动在江苏省无锡市建立了物联网研究院，重点开展 TD-SCDMA 与物联网融合的技术研究和应用开发。2010 年 1 月，国内正式成立了传感(物联)网技术产业联盟。工信部也宣布将牵头成立一个全国推进物联网的部际领导协调小组，以加快物联网产业化进程。2010 年 3 月，上海物联网中心正式揭牌。可见我国政府及各产业都对物联网非常重视，正在积极地进行物联网的研究开发和部署工作。

12.3.3 初探究竟——基于 EPC 模式的物联网简介

RFID 是物联网的基础，是物的信息获取的一个重要渠道。目前世界范围内推动 RFID 标准的组织是 EPC Global 和日本的 Ubiquitous ID。下面简单介绍一下 EPC 技术和基于 EPC 技术的物联网模型，初探究竟，看看物联网底层对物品信息的获取是如何实现的。

1. EPC 技术简介

EPC 的概念最早是在 1999 年由美国麻省理工学院 Auto-ID 中心提出的，后来在 2003 年由 EAN 和 UCC 成立了 EPC Global 来管理全球 EPC 标准和相关 EPC 的知识产权，并与 Auto-ID 中心合作管理 EPC 网络。

EPC 网络是在计算机因特网的基础上，利用射频识别、EPC 编码、因特网等技术，构造了覆盖世界上万事万物的物联网。EPC 系统是一个非常先进的、复杂的综合性系统。其最终目标是为每一物品建立全球的、开放的标识标准。EPC 系统主要由 3 大部分共 5 个基本要素组成，如图 12-14 所示。

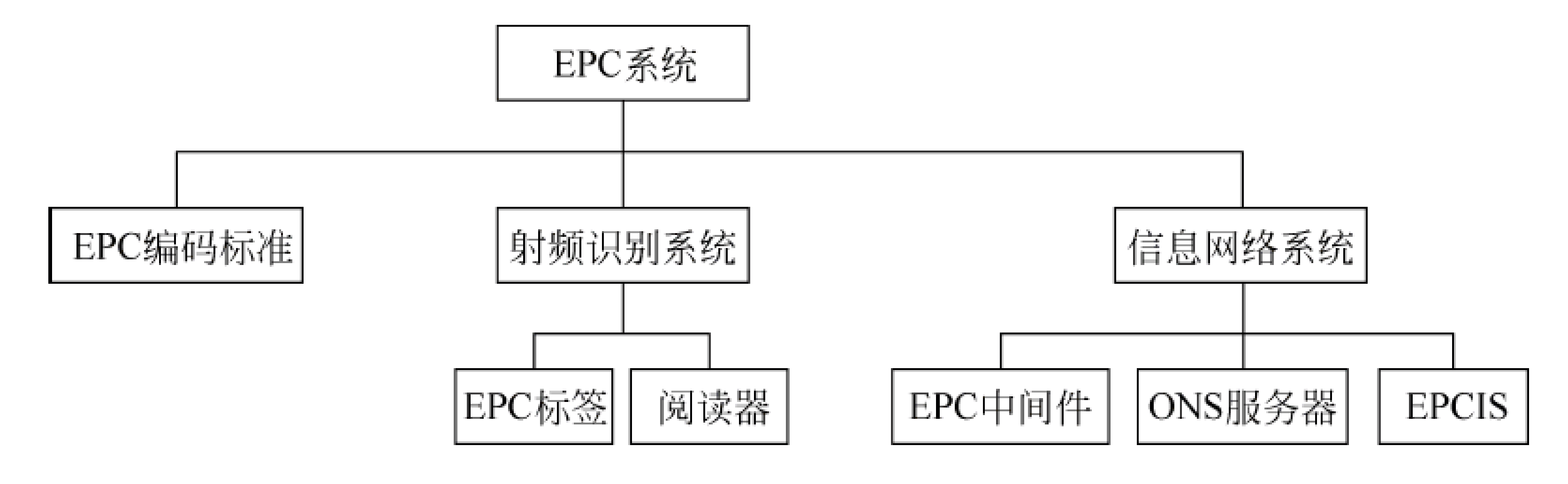


图 12-14 EPC 系统构成图

全球产品电子代码 EPC 编码标准是新一代的与 GTIN(全球贸易项目标识代码)兼容的编码标准，它是全球统一标识系统的拓展和延伸，是全球统一标识系统的重要组成部分，是 EPC 系统的核心与关键。该编码为使用协议的版本号、物品的生产厂商代码、物品的分类代码及单个物品的 SN 序列编号这 4 部分的数据字段所组成的一组数字。现在 EPC 标签的编码应用较多的主要有 64 位、96 位及 256 位三种。每个 EPC 编码具有全球唯一性，并且它的号码数量近似无穷大，足以分配到全球任一物品进行识别。

射频识别(RFID)系统实现 EPC 代码自动采集的功能模块，它由 EPC 电子标签、天线及阅读器组成。EPC 电子标签是产品电子序列号的载体，附着于可跟踪的物品上，从而实现全球流通。阅读器与信息系统相连，是读取标签中的 EPC 代码并将其输入网络信息系统的设备。EPC 标签与阅读器之间通过无线电感应方式进行信息交换。射频识别系统具有以下特点：非接触识别、可以识别快速移动的物品、可同时识别多个物品等。EPC 射频识别系统为数据采集最大限度地降低了人工干预，实现了自动化，是物联网不可或缺的重要

环节。

Auto-ID将Savant(EPC中间件)定义为一种软件技术,负责过滤、整合阅读器送来的标签或传感器的数据流,例如多个阅读器读取同一个EPC标签,则会在网络中产生多个相同的EPC序列号,Savant负责协调这些阅读器,过滤掉这些冗余信息,减少了传送到上层应用软件的数据量,降低网络的数据负荷。后来EPC Globe成立后将这部分重新命名为应用层事件,重新修整了与上层应用软件接口的定义。

ONS(对象名解析服务系统)可提供EPC查找服务,将给定的EPC代码转化为一个或多个含有物品信息的域名地址,以获取EPCIS服务器上更多的物品相关信息,其功能类似于因特网中的DNS。

EPC信息服务(EPCIS,原来称做PML服务)存放了大量制造商生产的所有物品相关数据信息的PML文件。PML是物体标记语言(physical markup language)的简称,它提供了一种通用的标准化词汇来表示EPC网络所能识别物体的相关信息,是在XML语言基础上发展起来的。

EPC系统的信息结构图如图12-15所示。

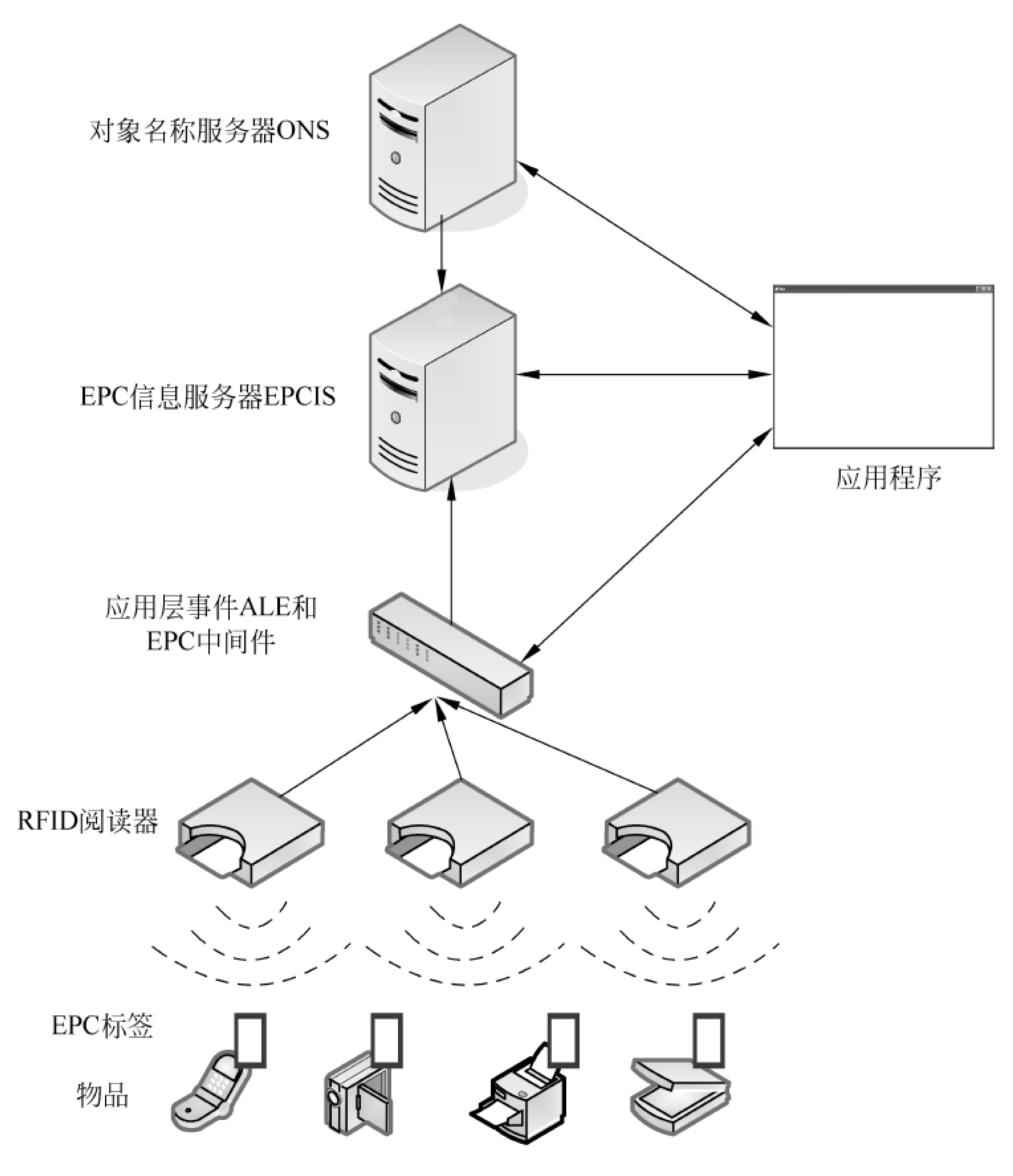


图12-15 EPC系统结构图

EPC 系统查询物品信息的基本流程如下：

① EPC 阅读器首先向标签发送查询命令，然后标签返回含 EPC 编码的信息。阅读器和标签之间的接口是射频接口。

② 阅读器将信息传送给 EPC 中间件。

③ 中间件对收集到的信息进行校对、过滤，生成事件序列，通过 ALE 接口传送给应用软件。

④ 应用软件向 ONS 服务器发送查询请求，ONS 服务器根据 EPC 编号，返回存有 EPC 编码对应物品原始信息的服务器的 IP 地址。

⑤ 应用软件通过 IP 地址就可以访问相应 EPCIS 服务器，获得该物品的信息。这个流程如图 12-16 所示。

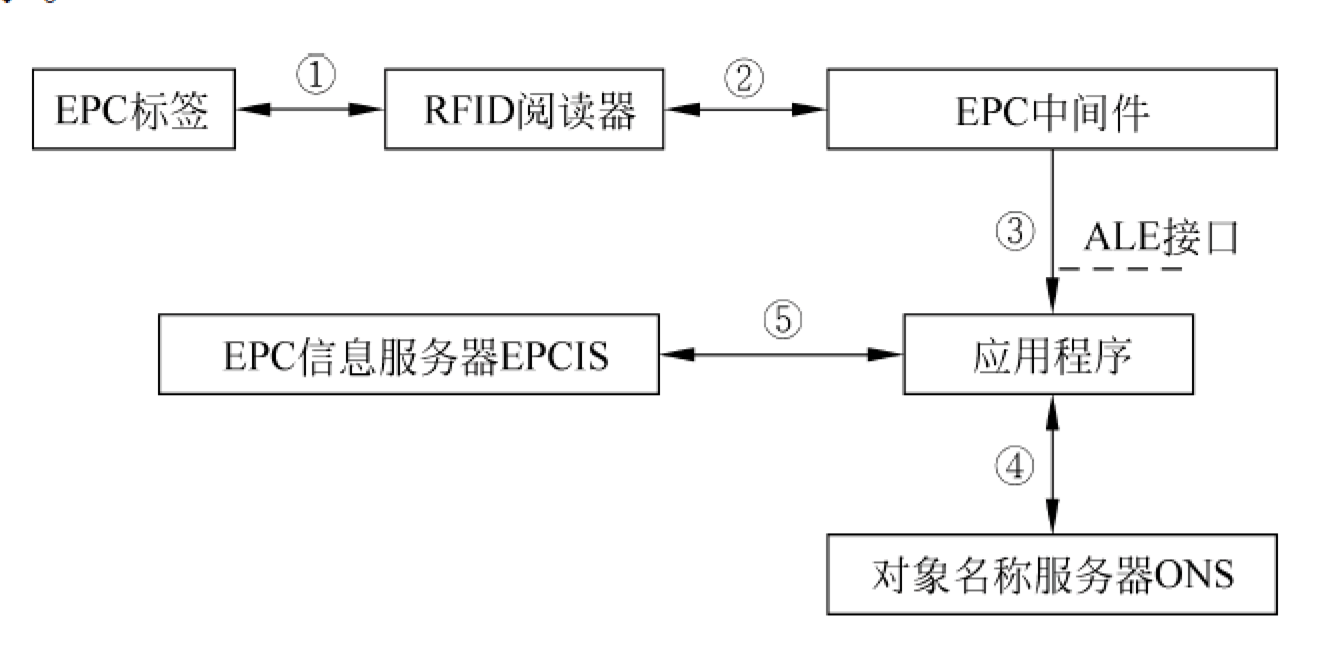


图 12-16 EPC 查询信息流程图

EPC 可以将全球商品唯一编码，使这些信息可以在网络中流通、共享。而且采 RFID 技术的标签和阅读器，与条形码相比自动化程度更高，可以减少商品在流通过程中的人工干预。目前，采用 EPC 标准的 RFID 技术已经在物流、零售、汽车、票务、交通、食品、航空（行李托管）等行业初具规模。

RFID 技术仅是物联网的基础，是实现物联网的第一步。仅有 EPC 模式的 RFID 系统还不能完全称做物联网。物联网的体系结构大致可以分为三层：感知层、传输层和应用层，如图 12-17 所示。ITU 的报告主要介绍的是感知层的四大关键技术——RFID 技术、传感器与传感器组网技术、小型化及纳米技术和智能技术。

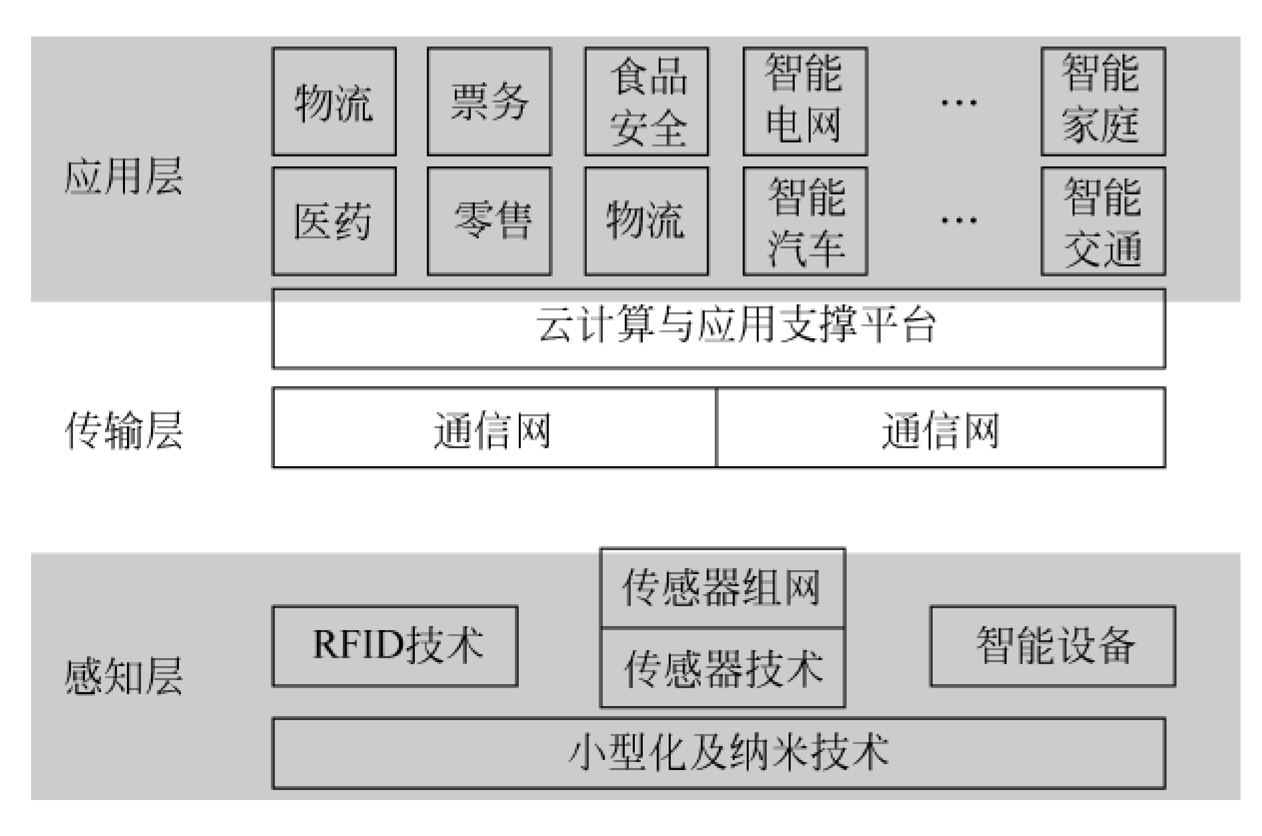


图 12-17 EPC 物联网体系结构

2. 基于EPC的物联网模型

物联网利用RFID技术和EPC标准(或其他标准)为每一个物品起了一个名字,而每个物品目前的状态、属性这些信息需要传感器采集,然后将传感器组网(请读者注意区分物联网中的传感器组网与时下也非常热门的无线传感网络技术的区别。物联网中的传感器是为了感知物的属性,传感器组网是为了采集已知的信息。而无线传感网络的目的是为了收集未知的信息,所以说二者目的不同,但是有一部分相同的技术基础),为每一个物品分配一个IP地址(需要未来的IPv6),使每一个物体不但拥有自己的身份ID,还有自己的地址。利用小型化及纳米技术,我们还可以将标签植入微小物体。

RFID、传感器、传感器组网、小型化及纳米技术和智能技术实现了"物联","网"可以理解为今天的通信网和因特网。通信网和因特网技术可以完成海量数据的传输、管理和存储,并基于网络平台开发各式各样的应用程序,展开丰富多彩的业务服务。由于物联网需要处理的数据量非常大,所以还可以携手云计算对海量的数据和信息进行分析和处理,对物体实施智能化的控制。

我们已经被今天庞大的因特网所震撼,但是跟物联网相比,因特网连接的都是相同类型的设备,只是性能有差异(例如PC、笔记本和手机等),而物联网要在同一个通信环境中连接不同功能、不同工艺、不同应用范围的对象,比因特网要复杂很多。

12.3.4 布满荆棘——浅析当前物联网发展瓶颈

ITU报告中为我们讲述的物联网是那么的让人向往。可惜现实很艰难,物联网的发展受到各方面诸多因素的阻挠,报告中小雅(报告原文是Rosa)的一天恐怕不会在几年内就变成读者您的一天。下面简单介绍一下目前物联网发展遇到的困难。

1. RFID技术问题

RFID技术的优越性毋庸置疑,但是目前RFID技术并没有完全取代原先的条形码技术,重要原因之一就是RFID的成本问题。虽然现在低频无源RFID标签的价格仅在20美分左右,但这对于低价的商品来说仍然太高。而且目前有些标签是没有安全措施的,这已经在业界饱受诟病,考虑到以后可能会在标签上采用一些轻量级的安全协议,需要增加额外的硬件开销,这又将增加RFID的价格。除了RFID成本的问题,还有就是RFID的技术适应性和可靠性并不让人十分满意,不同材质的物品需要不同的标签。条码的可靠性在99%以上。但RFID目前还远远达不到这个水平。对于木质、纸质的产品和农产品,RFID的可靠性在90%以上,但对于金属与液体产品,通常只能达到80%左右。可见目前国际先进的RFID技术水平离全球物联网的实现还是有一段距离的,仍需要多年的技术发展。具体到我国,低端RFID芯片生产已初具规模,但是高端RFID芯片制造水平比较落后,一些安全理念和国际先进的安全设计理念相比还有一定差距。而且目前基于超高频(UHF)的EPC系统的频段规定在860~960MHz之间,这个频段与我国多种无线通信的频段有些接近,如何在目前的UHF频谱范围内分配一个合适的频段并且保证超高频RFID与已有的无线通信之间不产生严重干扰是一个棘手的问题。目前我国有关UHF RFID频段分配的测试和试验工作正在进行之中。

2. 传感器能源问题

RFID技术是物联网的基础,通过RFID可以知道物品的"名字",光有"名字"还不够,我

们还需要知道物品的属性，这个功能是利用传感器实现的。物联网中的每个点还需要拥有数据交换和处理的能力，这是通过类似传感网的技术实现的。RFID和传感器都是物联网收集、交换物的信息的途径。目前，困扰国内外专家学者的是传感器网络中传感器的电源问题。现在传感网中一般的传感器寿命大概是几个星期的时间，而理想中物联网的传感器可以自发的收集或转化能源，这样传感网就可以持续运作，可见在供电技术上还有待新的突破。

我国领导对传感器和传感网技术大力支持，无锡拥有众多从事传感技术领域开发与研究的企业和高素质的学科带头人，取得了许多骄人的成绩，据专家分析目前我国在无线传感网络(这里再次指出物联网的传感器网络与无线传感网络是有差别的，但是有共通的技术基础)上与国外相比有同发优势，但是传感器产业化水平较低，在无线传感器技术领域尚无话语权，高端产品被国外厂商垄断。

3. 物联网安全与隐私问题

除了上述难题，物联网的安全与隐私问题也在面临着巨大的挑战。只有建立一套健全的安全与隐私保护机制，让用户能够安心地使用物联网，不必担心自己的个人信息安全受到威胁，才能实现物联网的全球化。目前，由于应用场合的原因，中低端RFID标签普遍缺少认证、通信加密等防护措施，安全性较低，这显然不能满足物联网的需要。但是为RFID标签增加安全机制就意味着成本的增加，所以，研究底价高安全性的RFID标签是一个难题。传感网节点通常情况下功能简单，使得它们无法拥有复杂的安全保护能力，而且传感网形式多种多样，温度、湿度、光感、压力等分布在各个不同的环境，数据的格式很难统一，传感网全部标准化十分困难，所以更无法建立一套标准的安全保护体系。智能是物联网的特性之一，尽量减少人工干预也是物联网预期目标中的一个，但是在一些场合缺少人的监管，很难防范对物联网的终端设备进行人为的机械性破坏。物联网中数据的传输和存储最后还是要依靠因特网，国外有学者分析，物联网与目前因特网中数据总量的疯狂增长会使目前的网络不堪重负，必须要找到一个新的存储和查找数据的方法，这也是威胁因特网安全的一个很棘手的问题。

物联网的安全与隐私问题不仅为威胁到个人，也会带来对国家安全的挑战，物联网的研究和建设需要和国外机构进行项目合作，确保企业商业机密、国家机密不被泄漏，这不仅是一个技术问题，而且还涉及到国家安全问题。未来物联网全球化部署时，保卫我国国家安全信息更是具有国防的战略意义，必须引起高度重视。

还有一个安全隐患就是RFID的电磁辐射。具最新研究资料表明，一般的RFID标签属于小信号辐射，不会对身体造成影响，无源UHF标签的读写器需要发送较大的射频功率以激活标签工作，并为其射频供电，成年人在辐射半径内短时间停留，人体所承受的电磁辐射完全在安全的限度之内。但是物联网的RFID标签及读写器无处不在，人体将很可能长时间被电磁辐射笼罩，这样究竟会不会威胁正常人的健康，尤其是婴儿，目前这个问题还需要进一步研究。

4. 物联网标准和专利问题

标准与专利问题也是阻碍物联网发展的拦路虎。标准的统一是实现物联网全球化的关键，没有类似因特网TCP/IP这样公开的被全球认可的协议，物联网不可能延伸到世界各个角落。但是今天关于RFID标准化的EPC Global、日本的Ubiquitous ID组织倡导的EPC

标准和 UID 标准不仅在通信频率上相差甚远,系统架构也区别很大,而且两个组织都不会放弃全球市场,很难在 RFID 标准化上达成妥协,两个组织的竞争关系会对将来 RFID 的发展造成很大的影响,从长远看也会威胁物联网的生存能力。因特网的大发展在 20 世纪 90 年代,当时人们对专利的意识比较淡薄。现在物联网的发展还处于萌芽阶段,但是已经有一些公司迫不及待地为自己的产品或技术申请专利保护,这就意味着未来物联网的开发商或运营商必须支付许多巨额的使用专利费用,这笔钱最终会转移到每个客户身上,所以一旦使用物联网的费用不能在一个合理的范围,物联网终将会成为富人的玩物,难以实现全球化。

5. 管理和商务问题

管理也是物联网全球化面临的一个重要障碍。谁来引领物联网的发展?谁来制定物联网的标准?如果缺少一个权威的组织负责统一规划物联网各个产业的标准,而像今天一样每个技术领域自由发展,很有可能会出现混乱、竞争等不统一的局面,就像 EPC 与 UID 的竞争关系。这个管理组织以何种形式存在,是在国家领导下的机构、联合国监管下的专题工作组,还是产业联盟?这个问题也是目前急需公开讨论的。这个管理组织不但要负责标准的规划,还要努力试图打破物联网中各产业的壁垒,促成各个子物联网的融合,避免各个子物联网形成信息孤岛。

目前物联网还没有出现清晰的商业模式,如何基于物联网建立合理的商业模式也是目前令许多业内人士困惑的问题。理想中的物联网是将全世界的物品连接起来,但是,所有物品都有商业价值吗?所有物品都有必要连入物联网吗?显然不是。例如一个绣花针、一个小气球,基本没有人会关心它们的 ID 号和 IP 地址,而这些数据可能在物联网中占大多数。试想我们的物联网中一大半数据都是冗余的,这些数据还要耗费海量的存储,这将大大降低物联网的有效利用率,浪费资源,开发商显然难以赞同这样的方案。物联网若想像因特网一样普及,还需要一些高质量的免费或价格非常便宜的应用服务软件,就像因特网的邮箱、搜索引擎、聊天工具和博客等受众多客户喜爱的服务。

上面简单地介绍了目前物联网发展中遇到的各方面难题。在现在看来,美好的物联网像是一座空中楼阁,漂在天上,离我们还很遥远;前方的路布满荆棘,各种困难犹如一座笔直的山峰矗立在我们面前,一步失足便会摔回原点。但是历史的经验告诉我们,困难终将敌不过人类的智慧,团结的力量会击碎一切拦路石,相信在各国精英的共同努力下,全球化的物联网可以突破重重困境,改善人类未来的生活!

12.3.5 继往开来——通往未来物联网世界的倒计时

欧洲智能系统集成技术平台(EPoSS)在 2008 年发布了一篇名为 *Internet of Things in 2020* 的报告,这篇报告的主要观点是由欧盟委员会和 EPoSS 邀请全球 80 多位相关领域的专家组成的专题小组的研究结果,向我们展示了未来全球化物联网的技术研究方向、在不同行业的应用设想和对社会及人类造成的影响。

在未来几年内,随着 RFID 和 IPv6 技术的广泛应用,大批数量的物品都拥有自己的 IP 地址,并且可以连接到网络,这时会迎来第一波物联网的高潮。此时全球化无缝连接的物联网将主要面临两个巨大的挑战:一个是现有的多种网络融合问题;另一个是物联网的“体积”问题,现在的 IT 产业并没有将上亿的对象连接到 IP 网络的经验。其他的难题(例如地

址约束、安全功能、声音和视频信号的多点传送等)都会被正在研究的技术解决。还有一个需要在物联网发展前期解决的问题就是相关法律法规的制定,这也是目前许多普通民众非常关注的问题。法律法规的尽早制定,可以最大限度地保障消费者的安全与隐私,使人们可以放心地享受物联网的各种应用和服务。

报告还推测了物联网技术发展在未来 20 年的时间表,如表 12-1 和表 12-2 所示。表 12-1 所示的进程是立足于现在的技术,推测未来 20 年的技术走向,可以看作是现有技术的演化;表 12-2 所关注的焦点是一些新的或需要加强的研究课题。这两个时间表以每五年为一个时间段,共分成了四个阶段,如果全球化物联网能够如报告所预期的顺利发展,那么我们就可以从现在开始为迎接未来的物联网世界倒计时。

表 12-1 正在进行的研究和未来技术趋势推测

	2010 年之前	2010—2015 年	2015—2020 年	2020 年以后
人们的预期	• RFID 被人们认可 • 意识到物联网的好处(食品安全、防伪、卫生保健) • 消费者担心隐私问题 • 改变人们的工作方式	• RFID 和阅读器无处不在 • 业务不断变化(工序、模型和工作方式) • 智能电器的使用 • 设置访问控制权限 • 新型零售业和物流业	• 半智能化 • 将 RFID、传感器和智能芯片集成 • 智能交通 • 能源和资源的保护	• 全智能化 • 掌握环境情报 • 现实世界和虚拟世界能相互作用 • 搜索物理世界(利用搜索引擎搜索实际物品) • 实现现实世界的虚拟化
政治与管理	• 存在管理方法 • 有关隐私的法规 • Address cultural barriers • 未来的因特网管理	• 欧盟管理 • 频谱管理 • 可持续能源消费指南	• 拥有身份验证、信任和核查机制 • 社会健康安全	• 拥有身份验证、信任和核查机制 • 社会健康安全
标准	• RFID 的安全和隐私标准 • 无线通信频段的分配	• 每个产业有明确的标准	• 交互标准	• 智能响应行为标准
技术应用的预期	• 连接物体 • RFID 技术应用于物流业、零售业和制药业	• 物体网络化 • 增加互操作性	• 物体半智能化,有执行能力 • 分布式代码 • 全球化应用	• 物体全智能化 • 人、物和服务的网络融合成一个 • 产业整合
器件	• 更小、更低价的电子标签、传感器、主动系统	• 提高信息容量和感知能力	• 超高速器件	• 更低价的材料 • 新的物理效应
功耗	• 低功耗芯片组 • 降低能量消耗	• 改善能量管理 • 提高电池性能	• 可再生能源 • 能够转化多种能源	• 能量捕获

表 12-2 新的或需要加强的课题

	2010 年之前	2010—2015 年	2015—2020 年	2020 年以后
人们的预期	• RFID 广泛应用 • RFID 被人们认可	• 集成对象 • 改善生活环境 • 生物识别标识 • 新的工业生态系统	• 形成物联网 • 智能生活 • 利用纳米技术的体内医疗设备 • 安全的生活	• 释放全部物联网潜能 • 人、计算机、物品连成一体 • 自动医疗保健
政治与管理	• 第一个全球性指导组织 • 标准化组织	• 第一个全球性管理组织 • 统一开放的互操作性	• 拥有身份验证、信任和核查机制	• 物联网将包含一切
标准	• 网络安全 • 特定传感网络 • 分布式控制和处理协议	• 交互式协议和频率 • 电源和故障恢复协议	• 智能设备互助	• 健康安全
技术应用的预期	• 低功耗和低成本 • 协议和频率的互操作性框架	• 集成的标签和传感网将无处不在 • 分布式控制和数据库 • 特定的混合网络 • 恶劣的环境下应用	• 全球化应用 • 自适应系统 • 分布式存储和处理	• 智能物品无处不在 • 异构系统
器件	• 智能多频带天线 • 更小、更低价的标签 • 高频标签 • 小型化嵌入式阅读器	• 扩展、高频的标签和阅读器 • 高传输速度 • 片上集成天线 • 与其他材料整合	• 具有执行能力标签； • 智能标签 • 具有自主工作能力的标签 • 具有协同工作能力的标签 • 采用新原料制成的标签	• 可生物降解器件 • 纳米功率处理组件
功耗	• 低功耗芯片组 • 超薄电池 • 电源优化系统(能源管理)	• 能量捕获(能源转化、光电) • 印刷电池 • 超低功耗芯片组	• 能量捕获(生物、化学、电磁感应) • 恶劣环境下发电 • 能源循环利用	• 可生物降解电池 • 无线电力传输

参考文献

[1] GlobalPlatform. Card Specification Version 2.2,2006

[2] GlobalPlatform Card Technology. Secure Channel Protocol 03，Card Specification V2.2-Amendment D，Version 0.80，2009

[3] Sun Microsystems. JavaCard(TM) Specification 2.2.2 FinalRelease，http://java.sun.com/products/javacard/specs.htmI，java_card_kit_2_2_2-fr-spec.zip，2006

[4] PC/SC Workgroup. Interoperability Specification for ICCs and Personal Computer Systems. Part 2: Interface Requirements for Compatible IC Cards and Readers,2005
[5] International Standard ISO/IEC 24727. Identification cards. Integrated circuit card programming interfaces,2006
[6] 郑青艳.移动办公系统的设计与实现.广州:华南理工大学硕士论文,2004
[7] 百度百科.移动办公[EB/OL],http://baike.baidu.com/view/718159.htm
[8] 赛迪网.Java 身份识别智能卡有望"一卡统天下",http://news.ccidnet.com/art/1032/20050808/305473_1.html,2008
[9] 董 威,杨义先.一种跨行业多应用智能卡系统模型及实现.计算机工程,2007,33(8):23~26
[10] 王同洋,秦保安,吴俊军.一卡多用安全管理平台.计算机应用,2005,25(1):154~159
[11] International Telecommunication Union. The Internet of Things,2005
[12] EPoSS. Internet of Things in 2020,2008
[13] 焦宗东.EPC 物联网中流通信息的研究.合肥:合肥工业大学,2007
[14] 黄小虎.基于 RFID 技术的 EPC 网络系统研究.广州:广东工业大学,2009
[15] 王俊宇,闵昊.EPC 系统结构及其面临的问题.小型微型计算机系统,2006,27(07):1280~1284
[16] 柳维长.实现全球物联网存在的几个问题.信息与电脑,2005,(05):10~13
[17] 荆继武,林璟锵,冯登国.PKI 技术.北京:科学出版社,2008
[18] 无线技术世界暨物联网国际高峰会议,2009
[19] 闫瑞芬.仁智相见:RFID 与物联网-行业精英观点聚焦.卡技术与安全,2009,(12):10~14
[20] RFID 世界网.http://www.rfidworld.com.cn
[21] 电子标签工作组.http://www.rfidgroup.org.cn
[22] RFID 中国网.http://www.rfidchina.org



Rijndael 算法 C++ 语言实现

```
////////////////////////////////////
/ *************************************
描述：Rijndael.h
      Rijndael 算法实现头文件
************************************* /
#if !defined(AFX_RIJNDAEL_H__65AE854C_7E77_4488_AC14_66161F67B341__INCLUDED_)
#define AFX_RIJNDAEL_H__65AE854C_7E77_4488_AC14_66161F67B341__INCLUDED_

//类型定义
typedef unsigned char          u1byte;
typedef unsigned short         u2byte;
typedef unsigned long          u4byte;
typedef signed char            s1byte;
typedef signed short           s2byte;
typedef signed long            s4byte;

//Windows 系统,数据存储为小端模式
#define LITTLE_ENDIAN

//AES 算法抽象类
class CAES
{
public:
    virtual void set_key(const u1byte key[], const u4byte key_bits) = 0;          //设置密钥
    virtual void encrypt(const u1byte in_blk[16], u1byte out_blk[16]) = 0;       //加密操作
    virtual void decrypt(const u1byte in_blk[16], u1byte out_blk[16]) = 0;       //解密操作
};

//32 位循环左移,循环右移
#ifdef _MSC_VER
#  include <stdlib.h>
#  pragma intrinsic(_lrotr,_lrotl)
#  define rotr(x,n) _lrotr(x,n)
#  define rotl(x,n) _lrotl(x,n)
#else
#define rotr(x,n)    (((x) >> ((int)(n))) | ((x) << (32 - (int)(n))))
#define rotl(x,n)    (((x) << ((int)(n))) | ((x) >> (32 - (int)(n))))
#endif

//4 字节数按字节颠倒位置 AB CD EF 01 -- -> 01 EF CD AB
#define bswap(x) (rotl(x, 8) & 0x00ff00ff | rotr(x, 8) & 0xff00ff00)
```

```
//从 int 数据中抽取字节
#define byte(x,n)    ((u1byte)((x) >> (8 * n)))

//整理成小端模式的 4 字节数据
#ifdef LITTLE_ENDIAN
#define u4byte_in(x)    (*(u4byte*)(x))
#define u4byte_out(x, v) (*(u4byte*)(x) = (v))
#else
#define u4byte_in(x)    bswap(*(u4byte)(x))
#define u4byte_out(x, v) (*(u4byte*)(x) = bswap(v))
#endif

//Rijndael 算法类
class CRijndael : public CAES
{
public:
    void set_key(const u1byte key[], const u4byte key_len);      //设置密钥
    void encrypt(const u1byte in_blk[16], u1byte out_blk[16]);   //加密操作
    void decrypt(const u1byte in_blk[16], u1byte out_blk[16]);   //解密操作
private:
    u4byte k_len;                                                //密钥长度,128,192,256
    u4byte e_key[64];                                            //加密轮密钥
    u4byte d_key[64];                                            //解密轮密钥
};
#endif // !defined (AFX_RIJNDAEL_H__65AE854C_7E77_4488_AC14_66161F67B341__INCLUDED_)
/////////////////////////////////////

/////////////////////////////////////
/***********************************
描述: Rijndael.cpp
Rijndael 算法实现源文件
***********************************/
#include "stdafx.h"
#include "Rijndael.h"

namespace
{
    u1byte pow_tab[256];        //幂数组,1,(x+1),(x+1)^2, (x+1)^3…用二进制表示
    u1byte log_tab[256];
    u1byte sbx_tab[256];        //S 盒
    u1byte isb_tab[256];        //逆 S 盒
    u4byte rco_tab[ 10];        //轮密钥扩展用
    u4byte ft_tab[4][256];
    u4byte it_tab[4][256];

    u4byte fl_tab[4][256];
    u4byte il_tab[4][256];

    u4byte tab_gen = 0;
```

```
#define ff_mult(a,b)    (a && b ? pow_tab[(log_tab[a] + log_tab[b]) % 255] : 0)

#define f_rn(bo, bi, n, k)                          \
    bo[n] = ft_tab[0][byte(bi[n],0)] ^              \
    ft_tab[1][byte(bi[(n + 1) & 3],1)] ^            \
    ft_tab[2][byte(bi[(n + 2) & 3],2)] ^            \
    ft_tab[3][byte(bi[(n + 3) & 3],3)] ^ *(k + n)

#define i_rn(bo, bi, n, k)                          \
    bo[n] = it_tab[0][byte(bi[n],0)] ^              \
    it_tab[1][byte(bi[(n + 3) & 3],1)] ^            \
    it_tab[2][byte(bi[(n + 2) & 3],2)] ^            \
    it_tab[3][byte(bi[(n + 1) & 3],3)] ^ *(k + n)

#define ls_box(x)                                   \
    ( fl_tab[0][byte(x, 0)] ^                       \
    fl_tab[1][byte(x, 1)] ^                         \
    fl_tab[2][byte(x, 2)] ^                         \
    fl_tab[3][byte(x, 3)] )

#define f_rl(bo, bi, n, k)                          \
    bo[n] = fl_tab[0][byte(bi[n],0)] ^              \
    fl_tab[1][byte(bi[(n + 1) & 3],1)] ^            \
    fl_tab[2][byte(bi[(n + 2) & 3],2)] ^            \
    fl_tab[3][byte(bi[(n + 3) & 3],3)] ^ *(k + n)

#define i_rl(bo, bi, n, k)                          \
    bo[n] = il_tab[0][byte(bi[n],0)] ^              \
    il_tab[1][byte(bi[(n + 3) & 3],1)] ^            \
    il_tab[2][byte(bi[(n + 2) & 3],2)] ^            \
    il_tab[3][byte(bi[(n + 1) & 3],3)] ^ *(k + n)

    //产生预置表格
    void gen_tabs(void)
    {
        u4byte i, t;
        u1byte p, q;

        // log and power tables for GF(2**8) finite field with
        // 0x011b as modular polynomial - the simplest prmitive
        // root is 0x03, used here to generate the tables
        for(i = 0,p = 1; i < 256; ++i)
        {
            pow_tab[i] = (u1byte)p; log_tab[p] = (u1byte)i;
            p = p ^ (p << 1) ^ (p & 0x80 ? 0x01b : 0);        // p = (x + 1)^i
        }
        log_tab[1] = 0; p = 1;

        for(i = 0; i < 10; ++i)
        {
            rco_tab[i] = p;
```

```
            p = (p << 1) ^ (p & 0x80 ? 0x1b : 0);
        }

        for(i = 0; i < 256; ++i)
        {
            p = (i ? pow_tab[255 - log_tab[i]] : 0); q = p;
            q = (q >> 7) | (q << 1); p ^= q;
            q = (q >> 7) | (q << 1); p ^= q;
            q = (q >> 7) | (q << 1); p ^= q;
            q = (q >> 7) | (q << 1); p ^= q ^ 0x63;
            sbx_tab[i] = p; isb_tab[p] = (u1byte)i;
        }

        for(i = 0; i < 256; ++i)
        {
            p = sbx_tab[i];

            t = p; fl_tab[0][i] = t;
            fl_tab[1][i] = rotl(t, 8);
            fl_tab[2][i] = rotl(t, 16);
            fl_tab[3][i] = rotl(t, 24);

            t = ((u4byte)ff_mult(2, p)) |
                ((u4byte)p << 8) |
                ((u4byte)p << 16) |
                ((u4byte)ff_mult(3, p) << 24);

            ft_tab[0][i] = t;
            ft_tab[1][i] = rotl(t, 8);
            ft_tab[2][i] = rotl(t, 16);
            ft_tab[3][i] = rotl(t, 24);
            p = isb_tab[i];

            t = p; il_tab[0][i] = t;
            il_tab[1][i] = rotl(t, 8);
            il_tab[2][i] = rotl(t, 16);
            il_tab[3][i] = rotl(t, 24);

            t = ((u4byte)ff_mult(14, p)) |
                ((u4byte)ff_mult( 9, p) << 8) |
                ((u4byte)ff_mult(13, p) << 16) |
                ((u4byte)ff_mult(11, p) << 24);

            it_tab[0][i] = t;
            it_tab[1][i] = rotl(t, 8);
            it_tab[2][i] = rotl(t, 16);
            it_tab[3][i] = rotl(t, 24);
        }     tab_gen = 1;
    }

#define star_x(x) (((x) & 0x7f7f7f7f) << 1) ^ ((((x) & 0x80808080) >> 7) * 0x1b)
```

```
#define imix_col(y,x)                    \
      u   = star_x(x);                   \
      v   = star_x(u);                   \
      w   = star_x(v);                   \
      t   = w ^ (x);                     \
      (y) = u ^ v ^ w;                   \
      (y) ^= rotr(u ^ t, 8) ^            \
      rotr(v ^ t, 16) ^                  \
      rotr(t,24)
}

#define loop4(i)                                          \
{   t = ls_box(rotr(t, 8)) ^ rco_tab[i];                  \
      t ^= e_key[4 * i];     e_key[4 * i + 4] = t;        \
      t ^= e_key[4 * i + 1]; e_key[4 * i + 5] = t;        \
      t ^= e_key[4 * i + 2]; e_key[4 * i + 6] = t;        \
      t ^= e_key[4 * i + 3]; e_key[4 * i + 7] = t;        \
}

#define loop6(i)                                          \
{   t = ls_box(rotr(t, 8)) ^ rco_tab[i];                  \
      t ^= e_key[6 * i];     e_key[6 * i + 6] = t;        \
      t ^= e_key[6 * i + 1]; e_key[6 * i + 7] = t;        \
      t ^= e_key[6 * i + 2]; e_key[6 * i + 8] = t;        \
      t ^= e_key[6 * i + 3]; e_key[6 * i + 9] = t;        \
      t ^= e_key[6 * i + 4]; e_key[6 * i + 10] = t;       \
      t ^= e_key[6 * i + 5]; e_key[6 * i + 11] = t;       \
}

#define loop8(i)                                          \
{   t = ls_box(rotr(t, 8)) ^ rco_tab[i];                  \
      t ^= e_key[8 * i];     e_key[8 * i + 8] = t;        \
      t ^= e_key[8 * i + 1]; e_key[8 * i + 9] = t;        \
      t ^= e_key[8 * i + 2]; e_key[8 * i + 10] = t;       \
      t ^= e_key[8 * i + 3]; e_key[8 * i + 11] = t;       \
      t = e_key[8 * i + 4] ^ ls_box(t);                   \
      e_key[8 * i + 12] = t;                              \
      t ^= e_key[8 * i + 5]; e_key[8 * i + 13] = t;       \
      t ^= e_key[8 * i + 6]; e_key[8 * i + 14] = t;       \
      t ^= e_key[8 * i + 7]; e_key[8 * i + 15] = t;       \
}

//设置密钥,key_len = 128,192,256
void CRijndael::set_key(const u1byte in_key[], const u4byte key_len)
{
      u4byte i, t, u, v, w;

      if(!tab_gen)
          gen_tabs();
```

```
        k_len = (key_len + 31) / 32;    //密钥长度整理成位的整数倍

        //密钥种子
        e_key[0] = u4byte_in(in_key);
        e_key[1] = u4byte_in(in_key +  4);
        e_key[2] = u4byte_in(in_key +  8);
        e_key[3] = u4byte_in(in_key + 12);

        //轮密钥扩展
        switch(k_len)
        {
        case 4:
            t = e_key[3];
            for(i = 0; i < 10; ++i)
                loop4(i);
            break;

        case 6:
            e_key[4] = u4byte_in(in_key + 16);
            t = e_key[5] = u4byte_in(in_key + 20);

            for(i = 0; i < 8; ++i)
                loop6(i);
            break;

        case 8:
            e_key[4] = u4byte_in(in_key + 16);
            e_key[5] = u4byte_in(in_key + 20);
            e_key[6] = u4byte_in(in_key + 24);
            t = e_key[7] = u4byte_in(in_key + 28);

            for(i = 0; i < 7; ++i)
                loop8(i);
            break;
        }

        d_key[0] = e_key[0];
        d_key[1] = e_key[1];
        d_key[2] = e_key[2];
        d_key[3] = e_key[3];

        for(i = 4; i < 4 * k_len + 24; ++i)
        {
            imix_col(d_key[i], e_key[i]);
        }

        return;
}

//1 - (Nr - 1)一轮,bo = byte out, bi = byte in, k = roundkey
#define f_nround(bo, bi, k)                   \
```

```
        f_rn(bo, bi, 0, k);                    \
        f_rn(bo, bi, 1, k);                        \
        f_rn(bo, bi, 2, k);                    \
        f_rn(bo, bi, 3, k);                    \
        k + = 4

//最后一轮
#define f_lround(bo, bi, k)                        \
        f_rl(bo, bi, 0, k);                          \
        f_rl(bo, bi, 1, k);                          \
        f_rl(bo, bi, 2, k);                        \
        f_rl(bo, bi, 3, k)

void CRijndael::encrypt(const u1byte in_blk[16], u1byte out_blk[16])
{
        u4byte b0[4], b1[4], * kp;

        //b0 数组是状态矩阵行陈列,b1 是计算后的中间状态的行阵列
        //              b0[0]
        // state =          b0[1]
        //              b0[2]
        //              b0[3]

        //轮密钥加
        b0[0] = u4byte_in(in_blk) ^ e_key[0];
        b0[1] = u4byte_in(in_blk + 4) ^ e_key[1];
        b0[2] = u4byte_in(in_blk + 8) ^ e_key[2];
        b0[3] = u4byte_in(in_blk + 12) ^ e_key[3];

        //轮密钥指针
        kp = e_key + 4;

        //如果是 Nk = 8
        if(k_len > 6)
        {
            f_nround(b1, b0, kp); f_nround(b0, b1, kp);
        }

        //如果是 Nk = 6
        if(k_len > 4)
        {
            f_nround(b1, b0, kp); f_nround(b0, b1, kp);
        }

        //如果是 Nk = 4
        //1 - 9 轮
        f_nround(b1, b0, kp);
        f_nround(b0, b1, kp);
        f_nround(b1, b0, kp);
        f_nround(b0, b1, kp);
        f_nround(b1, b0, kp);
```

```
        f_nround(b0, b1, kp);
        f_nround(b1, b0, kp);
        f_nround(b0, b1, kp);
        f_nround(b1, b0, kp);

        //最后一轮
        f_lround(b0, b1, kp);

        u4byte_out(out_blk, b0[0]);
        u4byte_out(out_blk + 4, b0[1]);
        u4byte_out(out_blk + 8, b0[2]);
        u4byte_out(out_blk + 12, b0[3]);
}

// decrypt a block of text
#define i_nround(bo, bi, k)              \
        i_rn(bo, bi, 0, k);              \
        i_rn(bo, bi, 1, k);              \
        i_rn(bo, bi, 2, k);              \
        i_rn(bo, bi, 3, k);              \
        k - = 4

#define i_lround(bo, bi, k)              \
        i_rl(bo, bi, 0, k);              \
        i_rl(bo, bi, 1, k);              \
        i_rl(bo, bi, 2, k);              \
        i_rl(bo, bi, 3, k)

void CRijndael::decrypt(const u1byte in_blk[16], u1byte out_blk[16])
{
        u4byte b0[4], b1[4], *kp;

        //b0 数组是状态矩阵行陈列,b1 是计算后的中间状态的行阵列
        //              b0[0]
        // state =          b0[1]
        //              b0[2]
        //              b0[3]
        b0[0] = u4byte_in(in_blk) ^ e_key[4 * k_len + 24];
        b0[1] = u4byte_in(in_blk + 4) ^ e_key[4 * k_len + 25];
        b0[2] = u4byte_in(in_blk + 8) ^ e_key[4 * k_len + 26];
        b0[3] = u4byte_in(in_blk + 12) ^ e_key[4 * k_len + 27];

        kp = d_key + 4 * (k_len + 5);

        if(k_len > 6)
        {
            i_nround(b1, b0, kp); i_nround(b0, b1, kp);
        }
        if(k_len > 4)
        {
            i_nround(b1, b0, kp);
```

```
            i_nround(b0, b1, kp);
        }
        i_nround(b1, b0, kp); i_nround(b0, b1, kp);
        i_nround(b1, b0, kp); i_nround(b0, b1, kp);
        i_nround(b1, b0, kp); i_nround(b0, b1, kp);
        i_nround(b1, b0, kp); i_nround(b0, b1, kp);
        i_nround(b1, b0, kp); i_lround(b0, b1, kp);

        u4byte_out(out_blk,      b0[0]); u4byte_out(out_blk + 4, b0[1]);
        u4byte_out(out_blk + 8, b0[2]); u4byte_out(out_blk + 12, b0[3]);
}
////////////////// End of Rijndael.cpp //////////////////////////
```



英文缩略语

缩略语	英 文 全 称	中 文 解 释
AA	Active Authentication	主动认证
ADF	Application Dedicated File	不包含下级目录的应用目录文件
AES	Advanced Encryption Standard	高级加密标准
AID	Application Identifier	应用标志符
APDU	Application Protocol Data Unit	应用协议数据单元
API	Application Programming Interface	应用程序接口
ATA	Answer To ATTRIB	ATTRIB 应答
ATC	Application Tracaction Count	应用交易次数
ATM	Automatic Teller Machine	自动柜员机
BAC	Basic Access Control	基本访问控制
BER	Basic Encoding Rules	基本编码规则
CBC	Cipher Block Chaining	密码块连接
CFB	Cipher Feedback	密码反馈
CHV	Card Holder Verification	持卡人验证
CICC	Close-Coupled Integrated Circuit Card	非接触式密耦合卡
CID	Card Identifier	卡标志符
CLA	Class Byte of the Command Message	命令报文的类别字节
COS	Chip Operating System	片上操作系统
CRC	Cyclic Redundancy Check	循环冗余校验
CryptoAPI	Cryptographic Application Programming Interface	加密应用程序接口
Cryptoki	Cryptographic Token Interface	密码令牌接口
CryptoSPI	Cryptographic Service Programming Interface	加密服务程序接口
CSP	Cryptographic Service Provider	加密服务提供者
CVM	Cardholder Verification Method	持卡人身份验证方法
DES	Data Encryption Standard	数据加密标准
DF	Dedicated File	专用文件
DFA	Differential Fault Analysis	差分错误分析
DPA	Differential Power Analysis	差分功耗分析
DSA	Data Signature Algorithm	数字签名算法
EAC	Extended Access Control	扩展访问控制
ECB	Electronic Code Book	电子密码本
ECC	Elliptic Curve Cryptosystem	椭圆曲线密码系统
ECDLP	Elliptic Curve Discrete Logarithm Problem	椭圆曲线离散对数问题
ECDSA	Elliptic Curve Digital Signature Algorithm	椭圆曲线数字签名算法

续表

缩略语	英 文 全 称	中 文 解 释
EDC	Error Detection Code	错误侦测编码
EF	Elementary File	基本文件
EOF	End of Frame	帧结束
ETU	Elementary Time Unit	基本时间单元
FCI	File Control Information	文件控制信息
FID	File Identifier	文件标识
GP	Global Platform	全球平台
IFD	Interface Device	接口设备
IFDSP	Interface Device Service Provider	接口设备服务提供者
ICAO	International Civil Aviation Organization	国际民用航空组织
ICC	Integrated Circuit Card	集成电路卡,IC 卡
ICCSP	Integrated Circuit Card Service Provider	IC 卡服务提供者
ISO	International Standardization Organization	国际标准化组织
JCRE	JavaCard Runtime Environment	Java 卡运行环境
JCVM	JavaCard Virtual Machine	Java 卡虚拟机
LC	Length of Command Data	终端发出命令域的实际长度
LE	Length of Response Data Expected	响应数据的最大期望长度
MFC	Microsoft Foundation Class	微软基础类库
MRZ	Machine Readable Zone	机读区域信息
MRTD	Machine Readable Travel Documents	机读旅行证件
NAD	Node Address	节点地址
NVB	Number of Valid Bits	有效位数目
OCR	Optical Character Recognition	光学字符识别
OSI	Open System Interconnect Reference Model	开放式系统互联参考模型
OFB	Output Feedback	输出反馈
P1	Parameter One	命令参数字节 1
PA	Passive Authentication	被动认证
P2	Parameter Two	命令参数字节 2
PCB	Protocol Control Byte	协议控制字节
PCD	Proximity Coupling Device	近耦合设备
PEM	Privacy Enhance Mail	增强型保密邮箱
PKCS	Public Key Cryptography Standards	公钥密码标准
PKI	Public Key Infrastructure	公钥基础结构
PICC	Proximity Integrated Circuit Card	非接触式近耦合卡
PIN	Personal Identification Number	个人身份号码
PUPI	Pseudo-Unique PICC Identifier	唯一 PICC 标志符
REQA	Request command，Type A	Type A 协议的请求命令
REQB	Request command，Type B	Type B 协议的请求命令
RFID	Radio Frequency Identification	射频识别
RFU	Reserved for Future Use	保留为将来使用
RSA	Ron Rivest，Adi Shamir and Leonard Adleman	一种非对称加密算法
RMI	Remote Method Invocation	远程方法调用

续表

缩略语	英 文 全 称	中 文 解 释
RTE	Runtime Environment	运行环境
SFI	Short File Identifier	短文件标识
SHA	Secure Hash Algorithm	安全散列算法
SPA	Simple Power Analysis	简单功耗分析
SIM	Subscriber Identity Module	用户身份模块
SW	Status Word	返回状态码
TLV	Tag-Length-Value	标签—长度—值
TR	Travel Record File	旅行记录文件
UI	User Interface	用户接口
UID	Unique Identifier	唯一标志符
VICC	Vicinity Integrated Circuit Card	非接触式疏耦合卡
WTX	Waiting Time Extension	等待时间扩展



智能卡应用测试工具介绍

智能卡 COS 系统质量的好坏直接影响着智能卡应用。如何对其进行高效完备的测试是 COS 开发者非常关注的事情。工预善其事，必先利其器。本附录介绍测试软件的目的就是为智能卡 COS 开发保驾护航，最大可能地保证了智能卡 COS 开发的正确性和功能完备性，是开发者手中必不可少的利器。

C.1 脚本解释器 FriMRTDScripter 7.6

C.1.1 软件介绍

本软件是为了测试机读旅行证件操作系统的通用指令和专用指令而开发的脚本解释器。它对 PC/SC 机具和非 PC/SC 机具都适用，实现了自动连接机具、自动记录脚本、自动记录运行结果、单步执行、循环测试等功能。

本软件运行在 PC 及其兼容机上，使用 Windows 操作系统。软件是绿色软件，无需安装，直接双击可执行文件 FriMRTDScripter.exe，就可以显示出软件的主界面，进行需要的软件操作。软件界面如图 C-1 所示。

整个用户界面分为 6 大部分，在图 C-1 中分别用 1～6 标注。这 6 大部分包括：

(1) 脚本编辑窗口。

(2) 结果显示窗口。

(3) 机具设置。

(4) 脚本编辑窗工具栏。

(5) 脚本执行工具栏。

(6) 信息栏。

C.1.2 软件运行

软件测试方法有多种，本测试软件与证件交互部分是以脚本形式实现的。将测试操作改用测试脚本来表述，脚本可以组合成用例，用例可组合成测试集，用例与测试集再统一到测试工程中管理，把测试脚本保存到文件，解决了重用问题。

软件内部实现一个脚本解释引擎，实现对脚本的解释，完成对读写机具的数据收发。脚本语言采用类 C 语言，具有以下特性：

(1) 支持变量定义和解释语句；

(2) 对变量支持算术操作符、赋值操作符、比较操作符、支持位于、位或、位异或等；

(3) 执行函数用 $ 前缀加函数名；

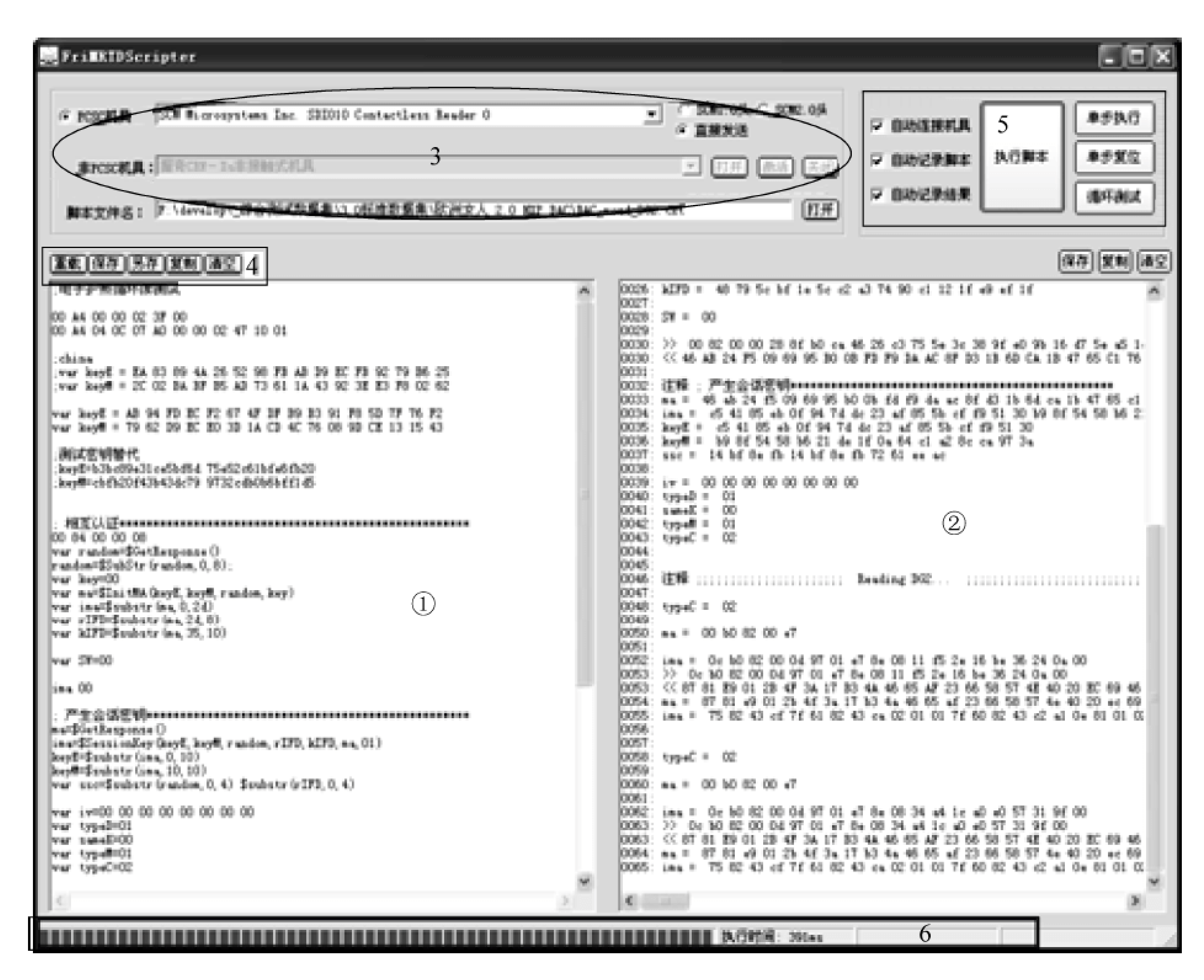


图 C-1 FriMRTDScripter 图形用户界面

(4) 支持 PC/SC 机具命令发送和接收；

(5) 支持 14443—3/4 透明指令；

(6) 支持条件语句、循环语句；

(7) 支持单步测试、循环测试；

(8) 支持文件＃include、脚本嵌套。

下面是相互认证命令的脚本和执行结果。

```
;电子护照循环读测试
var keyE＝AB94FDECF2674FDF B9B391F85D7F76F2
var keyM＝7962D9ECE03D1ACD 4C76089DCE131543
; 相互认证
00 84 00 00 08
var random＝$ GetResponse()
random＝$ SubStr(random,0,8);
var key＝00
var ma＝$ InitMA(keyE,keyM,random,key)
var ima＝$ substr(ma,0,2d)
ima 00
```

执行结果为：

```
0006: keyE ＝ AB94FDECF2674FDFB9B391F85D7F76F2
0007: keyM ＝ 7962D9ECE03D1ACD4C76089DCE131543
```

```
0009: 注释; 相互认证
0010: >> 00 84 00 00 08
0010: << 0C 6A A9 22 D0 D0 76 C3 90 00
0011: random = 0c 6a a9 22 d0 d0 76 c3 90 00
0012: random = 0c 6a a9 22 d0 d0 76 c3
0013: key = 00
0014: ma = 00 82 00 00 28 7f 90 fc 8b 12 bc af d0 e8 da c1 de 3c 24 f9 3b 49 98 43 ab 4b 05 70 76
26 a3 0b 8b 8f e3 5d 2b 82 11 78 15 22 32 58 07 36 6c 6f 62 bf 77 95 5d cd 8d cd 80 42 da 28 ec c7
2f bf a6 62 4a e9 f4
0015: ima = 00 82 00 00 28 7f 90 fc 8b 12 bc af d0 e8 da c1 de 3c 24 f9 3b 49 98 43 ab 4b 05 70 76
26 a3 0b 8b 8f e3 5d 2b 82 11 78 15 22 32 58 07
0020: >> (同上, 略)
0020: << (返回数据, 略)
```

C.1.3 系统配置

操作系统：Windows 2000/Windows XP
内存需求：>64MB
空间需求：>10MB
显示需求：>800×600 像素

C.2 安全测试软件 SecurityKit 1.6

C.2.1 软件介绍

公安部第一研究所自主开发的安全算法工具套件 SecurityKit 1.6 是依据国际标准实现的通用安全算法加解密计算的免安装软件。本软件适用于通用安全算法加解密运算/验证等场合，如安全应用软件、网络安全系统、法定证件(如电子护照和身份证)等。软件界面如图 C-2 所示。

安全算法工具套件软件 SecurityKit 1.6 主要功能有：流密码(如 RC4)加解密计算；对称算法(如 DES，3DES，AES 等)加解密计算；散列运算(SHA1、SHA224、SHA256、SHA384、SHA512、MD4、MD5)；大数运算；支持 RSA 密钥参数的生成、加解密、签名和验证；ECC 加解密、ECDSA 签名和验证；ISO 9797 规范的 MAC 码生成；Base64 编码和解码。

软件界面主要分成四个数据窗口：数据窗口 A、数据窗口 B、数据窗口 C 和数据窗口 R。前 3 个窗口主要放置操作数，计算得出的结果通常放置在数据窗口 R 中。

C.2.2 软件运行

菜单【设置】包括【低位在前】和【十六进制】两个选项，其中【低位在前】无效和【十六进制】有效为默认选项。界面图 C-3 所示。

菜单【对称算法】包括【DES/TDES】、【AES/Rijndael】、【ICAO Retail 3DES MAC】、【Full 3DES MAC】和【Full DES MAC】等 5 个子菜单。界面如图 C-4 所示。

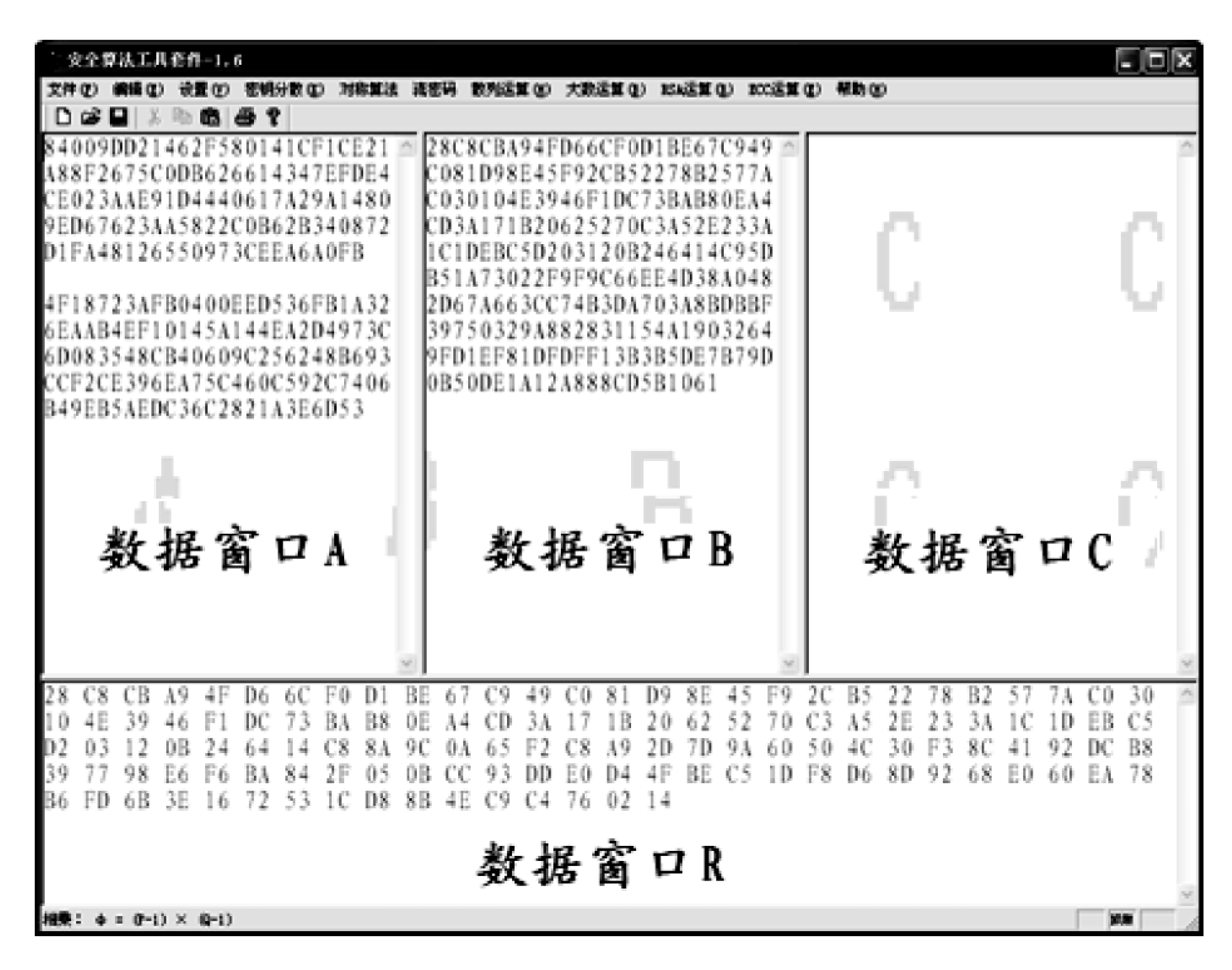


图 C-2　安全算法工具套件 SecurityKit 1.6 主界面

安全算法工具套件-1.6
文件(F)　编辑(E)　设置(V)　密钥分散(K)　对称算法　流密码　散列运算(H)　大数运算(B)　RSA运算(R)　ECC运算(E)　帮助(H)
低位在前
十六进制

图 C-3　【设置】菜单

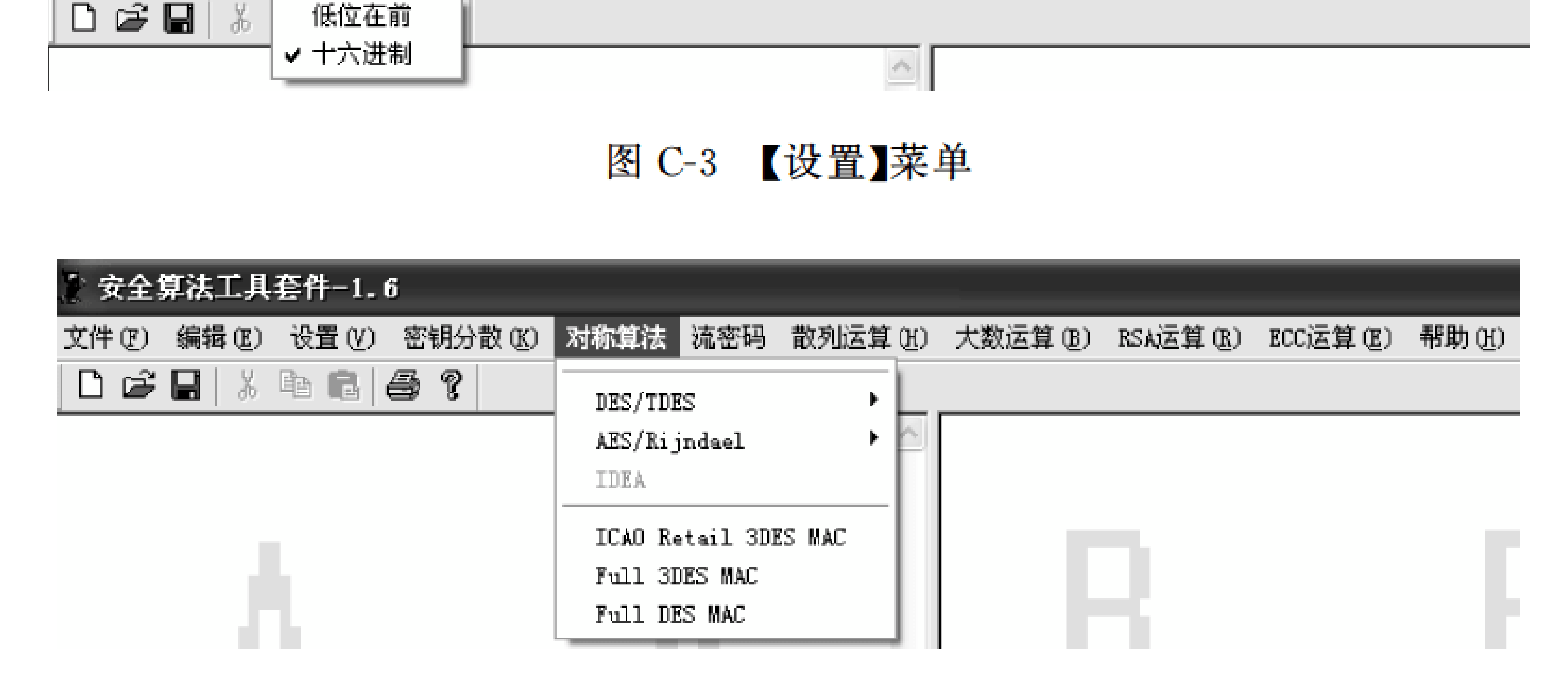


图 C-4　【对称算法】菜单

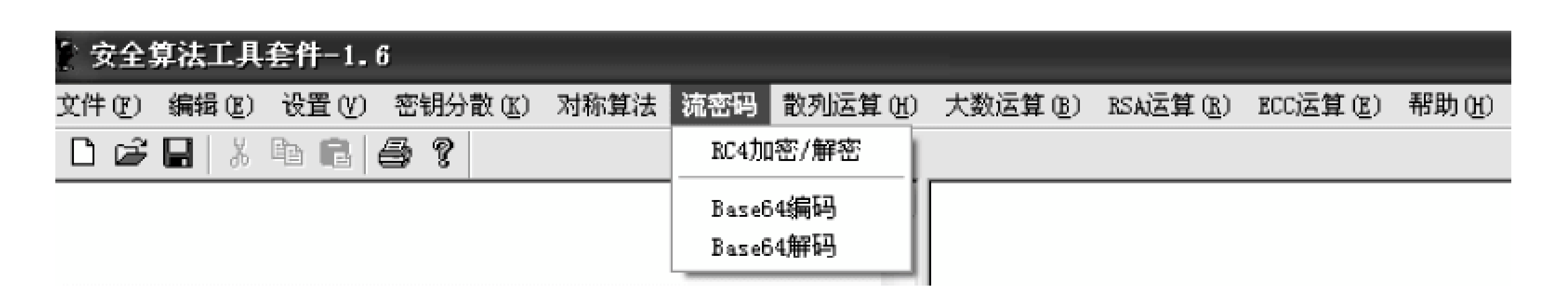
【DES/TDES】子菜单包括【ECB 加密】、【ECB 解密】、【CBC 加密】、【CBC 解密】四个选项。进行 DES/TDES 加/解密时，【数据窗口 A】为待加密的明文或待解密的密文。【数据窗口 B】为密钥，当密钥长度为 64 位时，密钥为 DES 密钥，采用 DES 加/解密；当密钥长度为 128 位时，密钥为 TDES 密钥，且 KEY1＝KEY3，采用 TDES 加/解密；当密钥长度为 192 位时，密钥为 TDES 密钥，采用 TDES 加/解密。【数据窗口 R】为加/解密输出。DES/TDES 算法参见“FIPS 46-3 DATA ENCRYPTION STANDARD”。

【AES/Rijndael】子菜单包括【128bit 加密】和【128bit 解密】两个选项。进行 AES/

Rijndael加/解密时,【数据窗口A】为待加密的明文或待解密的密文,且长度必须为128位的整数倍;【数据窗口B】为密钥且长度必须为128位;运算结果在【数据窗口R】中输出。

MAC运算采用本书第4章描述的算法进行计算。

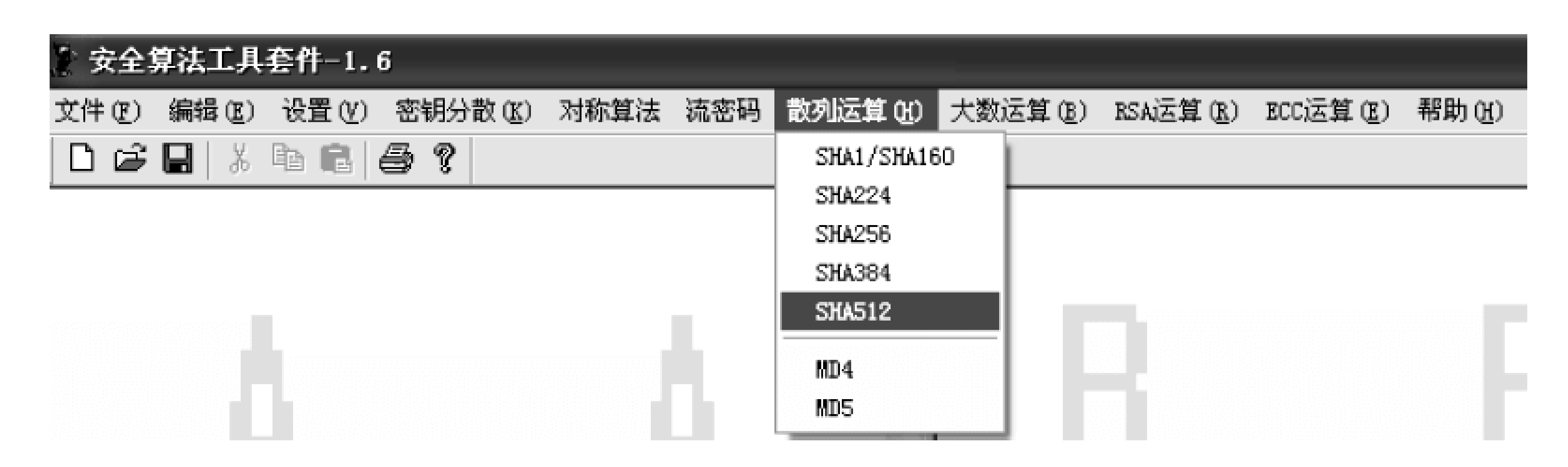
菜单【流密码】包括【RC4加密/解密】、【Base64编码】和【Base64解码】三个选项。界面如图C-5所示。



图C-5 【流密码】菜单

Base64编码将字符串ASCII码以6位长度分组,编码为模为64的编码格式。Base64编解码针对活动窗口中的内容。活动窗口即光标所在的【数据窗口A】、【数据窗口B】、【数据窗口C】或【数据窗口R】,下同。编解码后的内容在【数据窗口R】中输出。

散列运算用于计算活动窗口中的散列值。【散列运算】菜单包括如下选项:【SHA】、【MD4】和【MD5】。界面如图C-6所示。

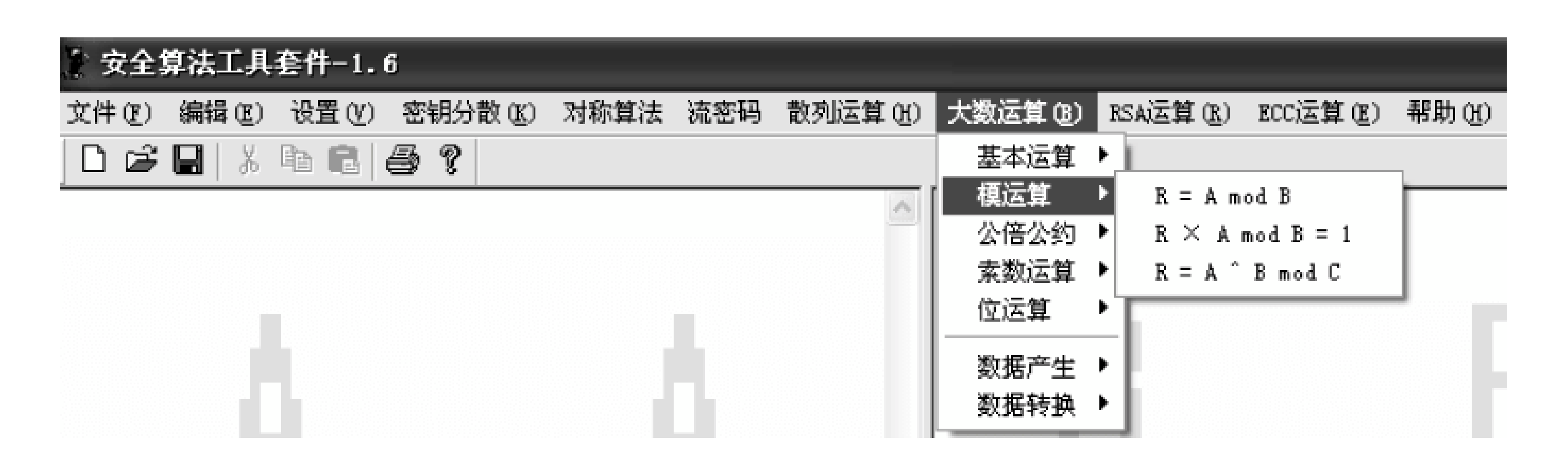


图C-6 【散列运算】菜单

SHA运算包括【SHA1/SHA160】、【SHA224】、【SHA256】、【SHA384】、【SHA512】五个子选项,分别对应160位、224位、256位、384位和512位的SHA运算。

MD4和MD5对活动窗口中的内容进行散列运算,得到128位的散列值。

菜单【大数运算】包括【基本运算】、【模运算】、【公倍公约】、【素数运算】、【位运算】、【数据产生】、【数据转换】七个子菜单。界面如图C-7所示。



图C-7 【大数运算】菜单

其中,【基本运算】包括大数的加减乘除;

【模运算】包括模运算、模逆运算和模幂运算;

【公倍公约】可以计算两个大数的最小公倍数和最大公约数；

【素数运算】包括判断给定的数是否是素数和产生给定长度的素数；

【位运算】包括按位与、或、异或运算；

【数据产生】可以产生 256 字随机数、产生 256 字节的 FF 和产生 256 字节的 00；

【数据转换】可以完成十六进制和十进制的相互转换。

菜单【RSA 运算】包括【生成 P 和 Q】、【计算 N＝P×Q】、【计算 Φ＝(P－1)(Q－1)】、【计算 D】、【计算 dP】、【计算 dQ】、【计算 iQ】、【SF 加密/签名验证】、【SF 解密/生成签名】九个选项。界面如图 C-8 所示。



图 C-8 【RSA 运算】菜单

菜单【ECC 运算】包括【参数设置】、【生成密钥对】、【加密计算】、【解密计算】、【ECDSA 签名(SHA1)】、【ECDSA 验证(SHA1)】、【ECDSA 签名】、【ECDSA 验证】八个选项。界面如图 C-9 所示。

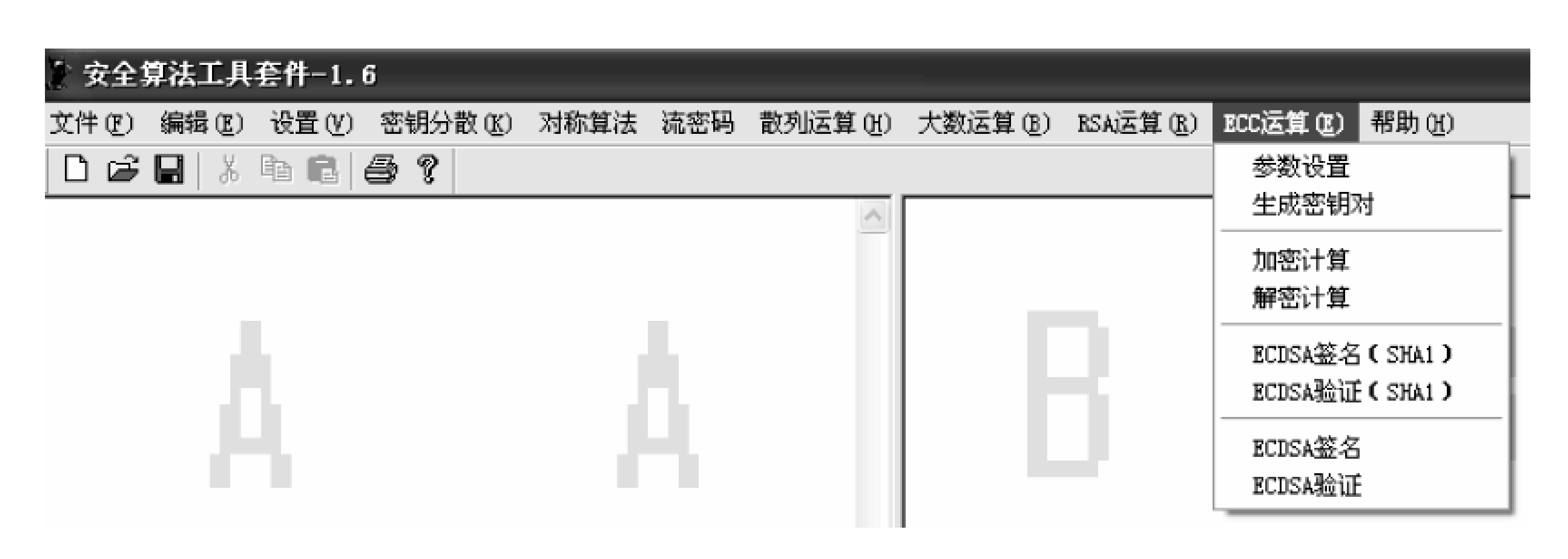


图 C-9 【ECC 运算】菜单

【参数设置】用于配置 ECC 参数，用于生成 ECC 密钥对；

【生成密钥对】通过配置后的参数和私钥，生成 ECC 公钥；

【加密计算】和【解密计算】用于 ECC 加解密；

【ECDSA 签名(SHA1)】和【ECDSA 签名(SHA1)】用于 ECDSA 签名与验证。这两个菜单内部已经将待签名消息进行了 SHA1 运算，而后面两个菜单则没有这一步骤。

【ECDSA 签名】和【ECDSA 签名】用于 ECDSA 签名与验证。

C.2.3 系统配置

操作系统：Windows 2000/Windows XP

内存需求：＞64MB

空间需求：>10MB

显示需求：>800×600 像素

C.3 电子旅行证件应用测试工具 FriMRTDcosEscort

C.3.1 软件介绍

公安部第一研究所自主开发的电子旅行证件应用测试软件 FriMRTDcosEscort.exe 依据国际民航组织（International Civil Aviation Organization，ICAO）的标准（RF PROTOCOL AND APPLICATION TEST STANDARD FOR E-PASSPORT - PART 3，Version：1.01，2007），测试电子护照是否符合 ISO/IEC 7816 规范和 ICAO 应用规范。软件界面如图 C-10 所示。



图 C-10 FriMRTDcosEscort 主界面

本软件利用通用 PC/SC 机具，针对电子旅行证件实施 ICAO 规范符合性测试，可逐条测试或者多项批量测试；本软件中实现了安全报文的封装和解析、测试用例的完备封装，支持明文方式或者基本访问控制(BAC)方式对电子旅行证件芯片进行访问；本软件可对用户测试条件进行配置，实时显示测试命令和响应，并支持测试报告输出等功能。

C.3.2 软件运行

本软件运行在 PC 及其兼容机上，使用 Windows 操作系统；软件无需安装，直接双击相

应可执行文件，即可显示出软件主界面，进行需要的软件操作。

本软件的使用步骤如下：

(1) 双击 FricosEscort.exe，启动本软件。

(2) 单击机具设置，选择使用的机具，常用的是 SCM Microsystems Inc. SDI010 Contactless Reader 0。

(3) 确认待测卡片中的 MRZ 信息是否为 MRZ 信息显示窗中显示的信息。

(4) 单击鼠标左键，选择测试用例，并根据该用例选择 Profile 中的访问控制方式以及 EF 文件。例如：Test Case 7816_B_5，该用例要求是 BAC 方式，并且选中 DG3(具体要求可参见【帮助】|【内容】)；运行此用例之前，要激活 DF 的 ICAO 状态，并在 Profile 中选中 BAC 方式和 DG3。

(5) 单击“执行”按钮，或者双击选中用例，执行该用例。

(6) 单击“统计”按钮，统计本次测试的测试结果，该结果会显示在软件底部的信息窗中。

(7) 可读取卡片内数据。自行开发的电子护照数据读取软件的界面和操作结果如图 C-11 所示。



图 C-11 卡片数据读取显示

特别说明：

(1) 选中用例总条目“Test Case 7816”，单击“执行”按钮，可执行所有用例。

(2) 选中一级子条目，例如“Test Case 7816_A”，单击“执行”按钮，可执行该一级子条目下所有用例。

(3) 运行图示说明如图 C-12 所示。

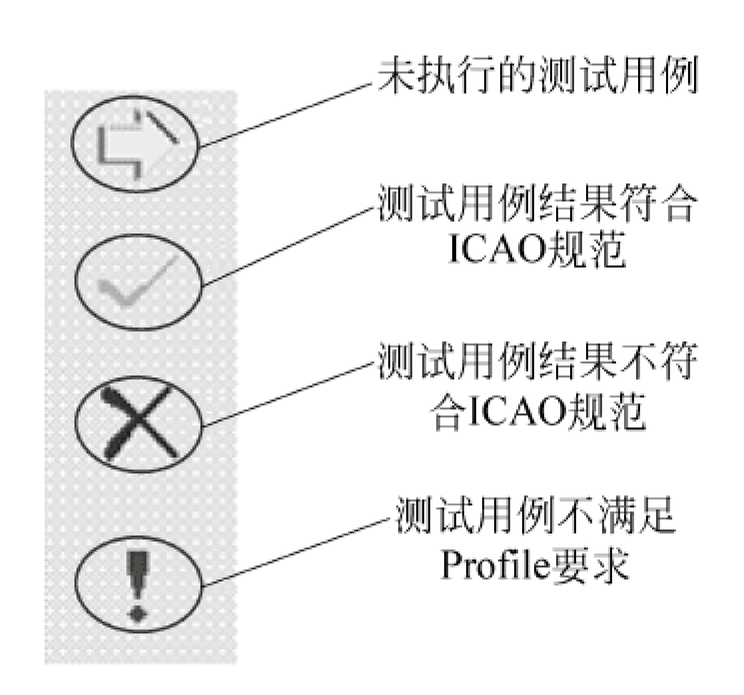


图 C-12 FriMRTDcosEscort 测试用例执行结果图示

C.3.3 系统配置

操作系统：Windows 2000/Windows XP

内存容量：至少 64MB

存储空间：至少 10MB

显示需求：>800×600 像素

C.4 电子护照综合测试软件 AllTest 7.6

C.4.1 软件介绍

世界各国电子护照的研究和试验工作如火如荼地进行，目前全球约 80 多个国家相继启动或者宣布使用电子护照计划。

电子护照是在普通护照上增加智能卡芯片制作而成，可存储姓名、性别、生日和出生地等个人信息以及指纹、脸部图像和虹膜等生物特征信息，是加强出入境管理、简化出入境手续、加快通关速度、防止偷渡、预防外来恐怖活动的重要技术手段。

公安部第一研究所承担了中国电子护照 COS 系统的开发。为保证电子护照 COS 开发的正确性，复核电子护照的业务流程的有效性和安全性，开发一套完备有效的测试软件是非常必要的。

电子护照综合测试软件 AllTest 7.6 软件界面如图 C-13 所示。

该软件的主要功能包括：

(1) 脸部/指纹等生物特征图像文件自动转换成 DG(data group)文件；

(2) 安全对象数据(document security object，SOD)文件自动生成；

(3) 公私钥文件 DG15/PriKey 自动生成；

(4) DES、3DES、RSA、SHA1 等加解密算法计算和验证功能，支持大数计算功能；

(5) 软掩模卡片下载功能；

(6) 脚本执行结果的统计、分析等处理功能。

主界面窗口为分割窗口方式，左右窗口分别为操作窗口 1、操作窗口 2。光标所在窗口为

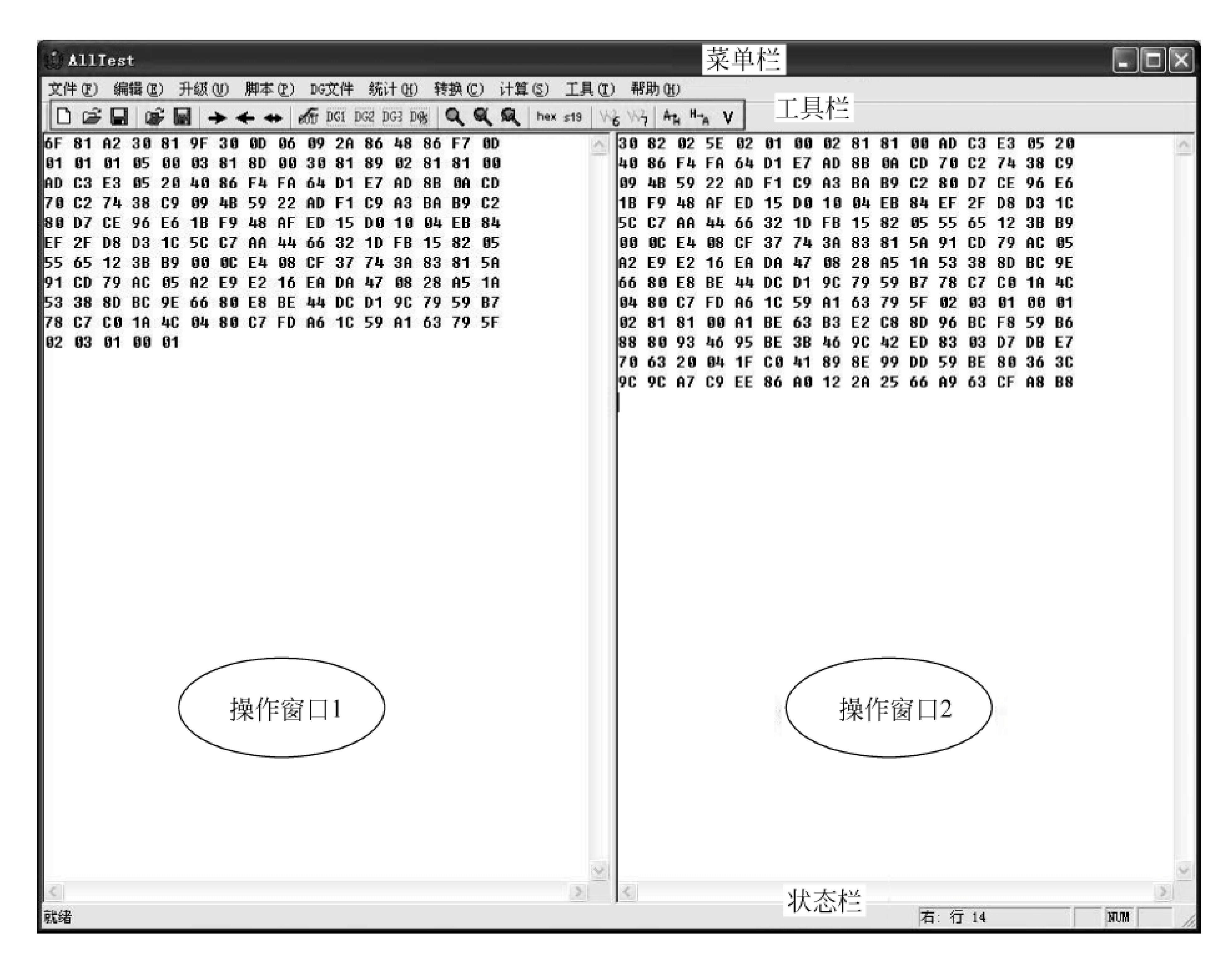


图 C-13 AllTest 主界面图

活动窗口。菜单栏中包括所有本软件支持的功能操作。工具栏包括本软件常见功能操作。

状态栏提示信息帮助或者操作结果。

图 C-14 显示工具栏提供的便捷功能按钮。其中打开和保存功能分为两种：文本方式和十六进制方式。这两种方式可处理所有文件。

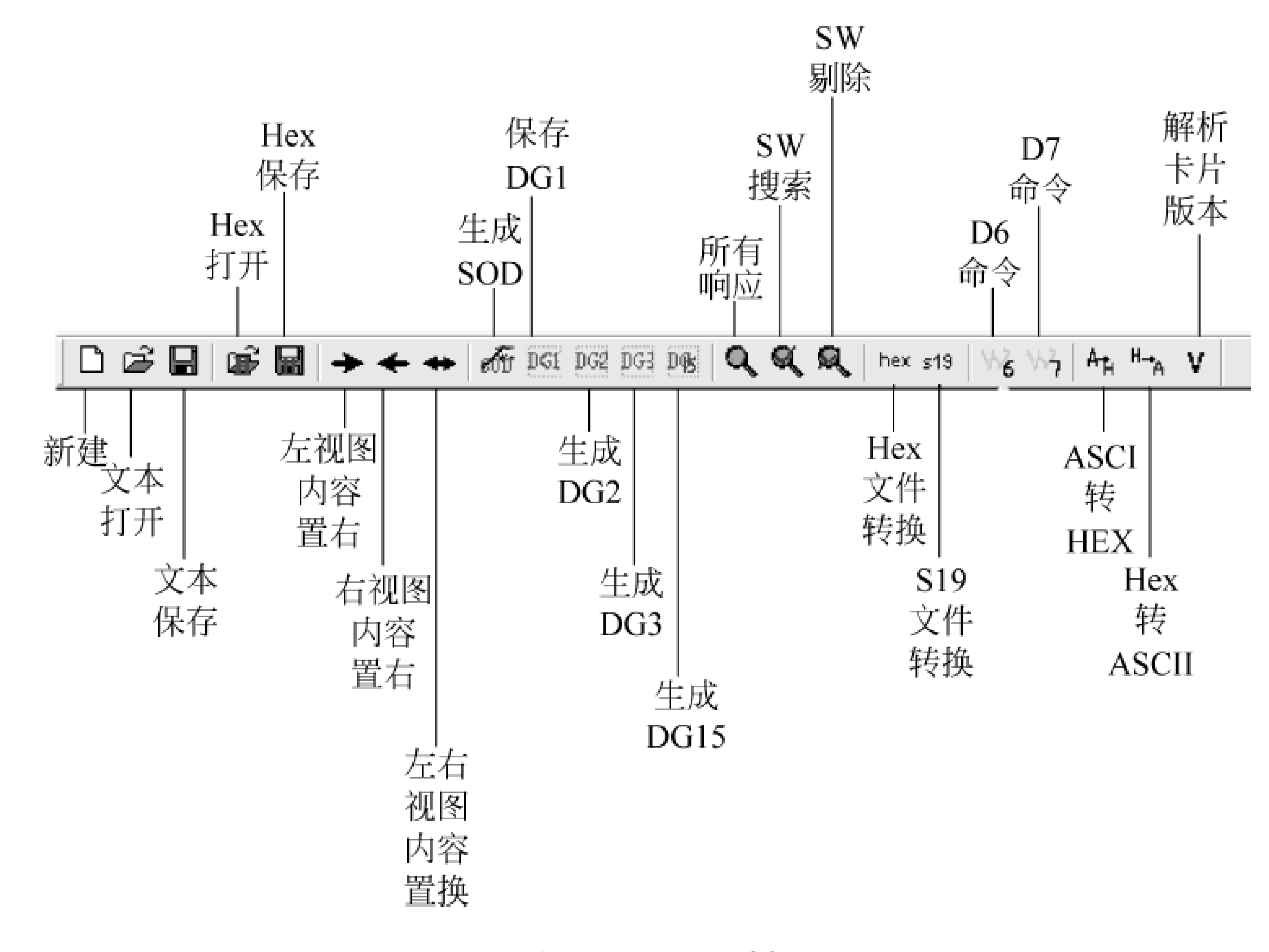


图 C-14 工具栏

C.4.2 逻辑数据封装

电子护照中存储的工作文件都是二进制文件，逻辑数据结构（logical data structure，LDS）如下所示。

（1）EF-COM。文件记录当前应用的通用信息，比如包括哪些 DG 文件。

（2）EF-SOD。每个 DG 文件计算 SHA1 值，然后经签名得到的文件。

（3）EF-DG1。机读区信息文件，即护照上光学可识别码（optical character recognition，OCR）。

（4）EF-DG2。脸部图像信息文件。

（5）EF-DG3。指纹图像信息文件。

（6）EF-DG4。虹膜图像信息文件。

（7）EF-DG15。公钥文件。

C.4.3 SOD 签名文件生成

被动认证（PA）在 ICAO 规范中是强制执行的。被动认证通过证件签发机关私钥来控制签发权限，利用哈希值保证证件中文件的完整性。

证件安全对象（SOD）文件存储 DGx 文件的 SHA1 值的签名。签名使用的证书为国家证件签名证书。

本软件生成 SOD 文件的步骤如下描述：

（1）单击工具栏中的按钮 455，显示 SOD 产生器对话框，如图 C-15 所示。

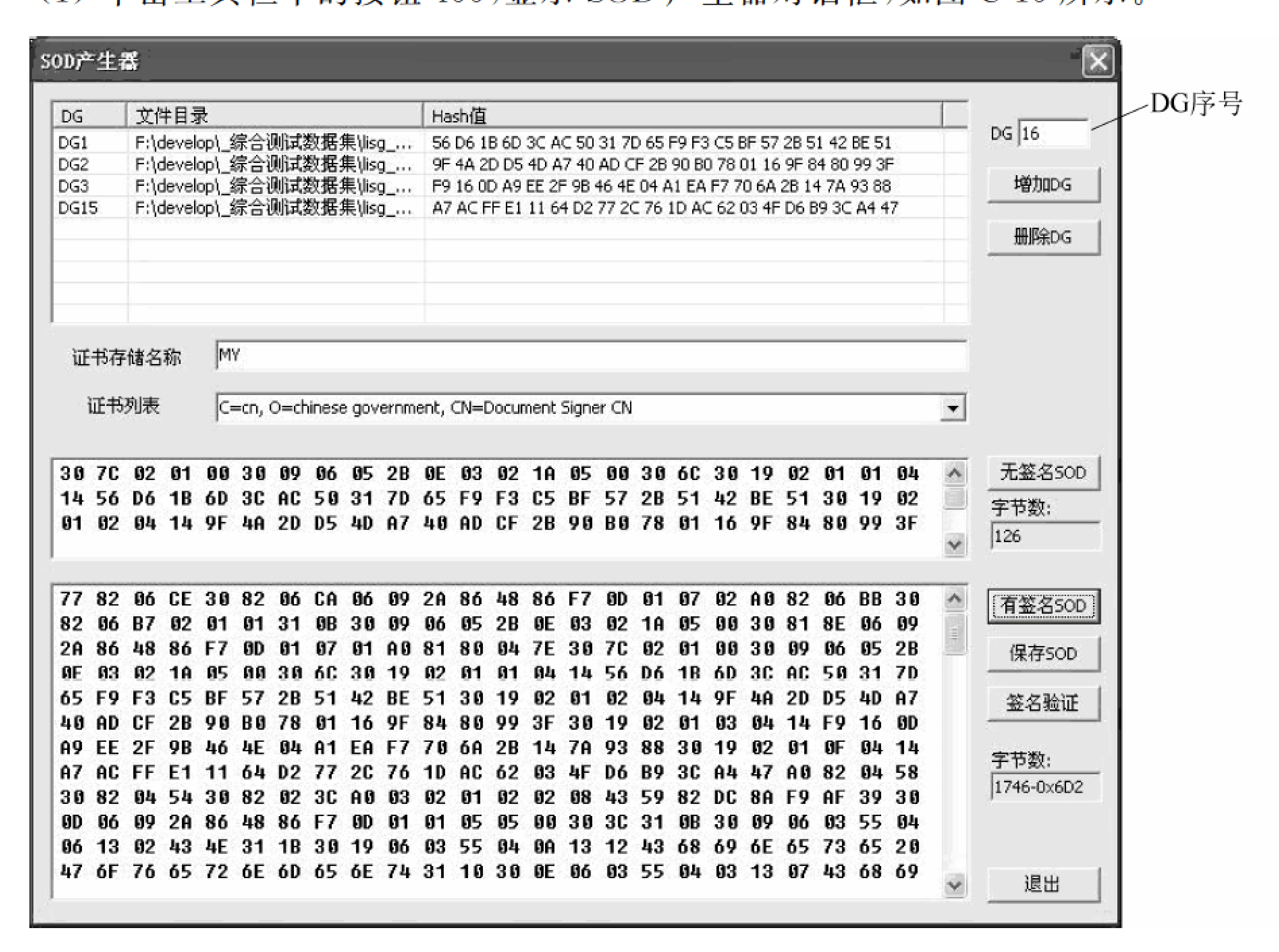


图 C-15 SOD 签名文件生成界面

(2) 先填写 DG 的序号,然后单击对话框中的【增加 DG】按钮,选择 DGx 文件,通常为 DGx.bin 文件。

(3) 再重复(2)操作,支持任意个 DGx 文件。注意 DG 的序号一定要对应。该 DG 序号默认为递增,如果 DGx 文件序列为 1,2,3,7,15,则需要修改 DG 序号框。

(4) 填写好 DG 文件后,选择证书列表中的国家证件签名证书,然后单击【无签名 SOD】按钮,再单击【有签名 SOD】按钮,可生成 SOD 文件内容。可单击【保存 SOD】按钮进行保存,以备后用。

C.4.4 RSA 计算

RSA 密码系统的安全性依赖于大数分解的难度,一般建议用户选择的素数 p 和 q 至少为 100 位,则 $n=pq$ 是至少为 200 位的十进制数。

在菜单【计算】中,有 4 项关于非对称密码算法 RSA 相关的功能,分别为:密钥对;PKSC8 格式;导出 PFX 文件;导出私钥。

C.4.5 其他功能

(1) 可解析 Infineon 卡片和 ST 卡片升级文件 *.HEX 和 *.S19,转换成标准的写命令,下载到 EEPROM 中进行升级操作。

(2) 可解析并下载软掩膜 COS 代码到 Infineon 卡片 SLE66P** 系列软掩膜卡片中。软件界面如图 C-16 所示。



图 C-16 软掩膜下载工具界面

(3) 菜单【工具】|【谐振频率】功能计算天线设计中的参数。根据公式 $f=\dfrac{1}{2\pi\sqrt{LC}}$,任意输入 f、L、C 三者中的两者,计算得出第三者。

(4) 菜单【工具】|【SW 查询】可根据输入状态码 SW(两字节 Hex 格式),单击【查询】按钮即可检测出错误信息。

(5) 根据 ISO/IEC 14443-3 中指定的 CRC-16 计算公式,输入待计算的字符串,单击【计算 CRC】按钮即可计算出 CRC_TYPEA 和 CRC_TYPEB。输入的字符串为十六进制串格式,忽略空格。软件界面如图 C-17 所示。

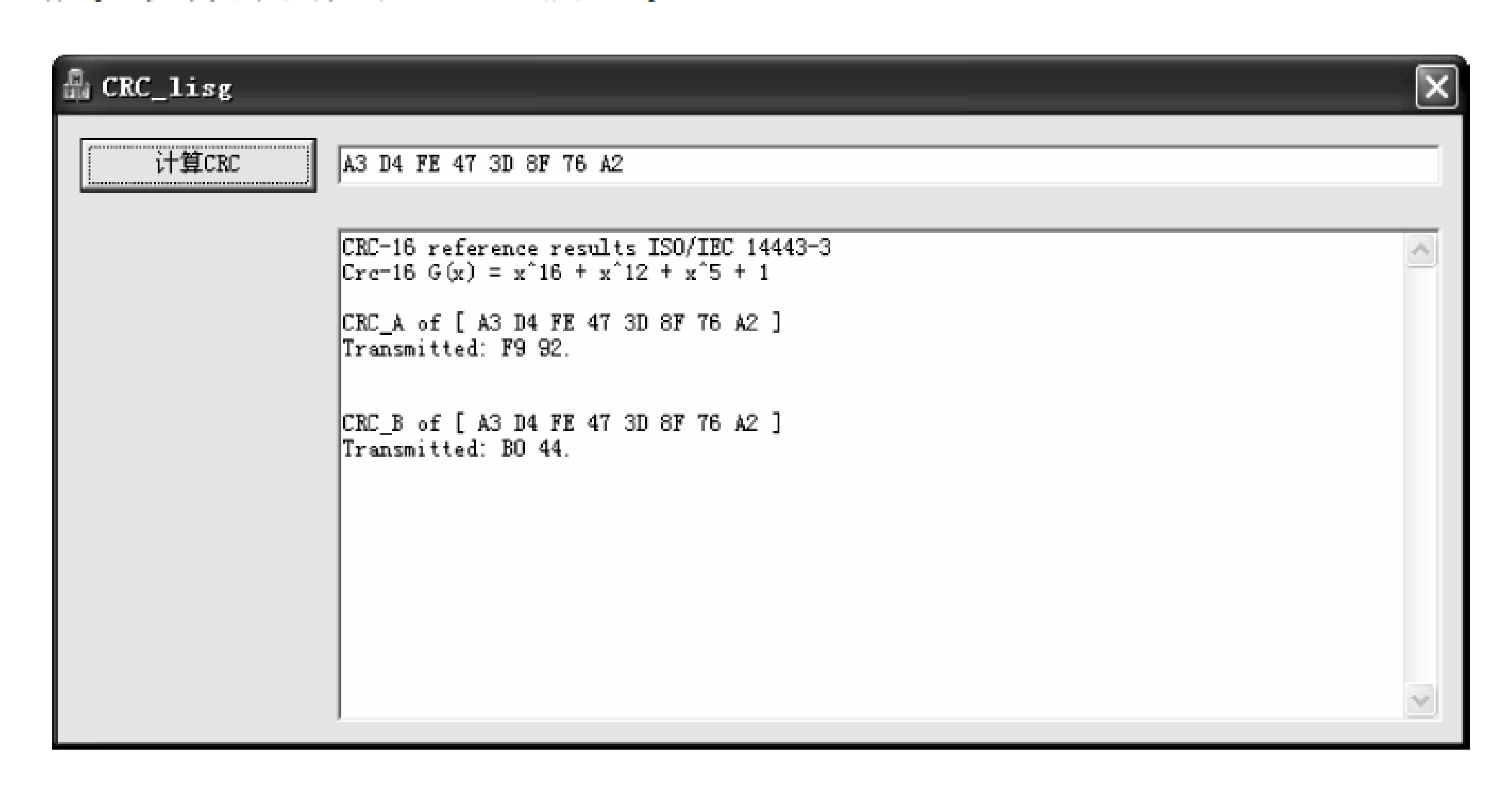


图 C-17　CRCA、CRCB 计算工具界面

(6) 对卡片产生的随机数进行真随机数测试。软件界面如图 C-18 所示。



图 C-18　真随机数测试工具界面

C.4.6 系统配置

操作系统：Windows 2000/Windows XP
内存容量：至少 64MB
存储空间：至少 10MB
显示需求：>800×600 像素